

Integrating Data and Model Space in Ensemble Learning by Visual Analytics

Bruno Schneider^{}, Dominik Jäckle^{}, Florian Stoffel, Alexandra Diehl, Johannes Fuchs^{}, and Daniel Keim

Abstract—Ensembles of classifier models typically deliver superior performance and can outperform single classifier models given a dataset and classification task at hand. However, the gain in performance comes together with the lack of comprehensibility, posing a challenge to understand how each model affects the classification outputs and from where the errors come. We propose a tight visual integration of the data and the model space for exploring and combining classifier models. We introduce an interactive workflow that builds upon the visual integration and enables the effective exploration of classification outputs and models. The involvement of the user is key to our approach. Therefore, we elaborate on the role of the human and connect our approach to theoretical frameworks on human-centered machine learning. We showcase the usefulness of our approach and the integration of the user via binary and multiclass classification problems. Based on ensembles automatically selected by a standard ensemble selection algorithm, the user can manipulate models and alternative combinations.

Index Terms—Classification, ensemble learning, data visualization, graphical user interfaces

1 INTRODUCTION

GIVEN a set of known categories (classes), *Classification* is defined as the process of identifying to which category a new observation belongs [36]. In the context of machine learning, classification is performed on the basis of a training set that contains observations whose categories are known. A key challenge in classification is to improve the performance of the classifiers, hence new observations are correctly assigned to a category. Classification can be performed with a variety of different methods tailored to the data or the task at hand. Examples include, among others, decision trees, support vector machines, or neural networks.

Research proposes to improve the accuracy of classification using *Ensemble Learning* [12], [44], also known as *Multiple Classifier Systems* (MCS) [34]. Such systems suggest combining different classifiers, potentially expanding the space of representable functions by using distinct learning philosophies at the same time. Well-known approaches for building ensembles propose to either train the same model successively with different subsets of the data [5], [14], to combine different model types [21], [41], or to combine different strategies such as bagging [5] with random feature combinations in Random Forests [6]. Generally speaking, the application of ensembles increases the complexity of the classification process bringing in the inherent problem of decreasing comprehensibility. In particular, it is challenging to understand how and to what extent the models contribute to the classification, as well as which models produce a significant number of classification errors.

Visual and automatic methods for the analysis of Classification outputs in Ensemble Learning do not provide a direct link from the data space back to classification model spaces with other candidates for experimenting with new ensemble configurations. Regarding the visual methods, they also do not scale properly to represent a greater number of classifiers in ensemble model spaces. For example, in [39] Silva and Ribeiro show how the models contribute individually, but the analysis is limited to inspect the ensemble after making the decision of which models will take part on it. In [40], Talbot et al. present a system in which is possible to interact and combine models and their classification outputs through *confusion matrices*, but with a limited set of model candidates. However, we can build classification data spaces and connect them with model spaces covering a wide range of the parameter space for the classification problem at hand. By linking model and data spaces, we foster an analysis process with a feedback loop that allows the effective exploration of these spaces driven by the user notion of importance. To the best of our knowledge, this workflow is not supported by any visual or automatic method for analyzing and exploring ensembles of classifiers.

In this work, we aim to address the research question: *How to integrate data and model space to enable visual analysis of classification results in terms of errors in Ensemble Learning?*

We propose an interactive visual approach for the exploration of classification results (data space) in close integration with the model space. Its main goal is to give direct access to models in classifier ensembles, thus enabling to experiment with alternative configurations and seek for local classification patterns that are not visible through aggregate measures. We visualize each classified data point and then provide direct access to each individual model that is part of the ensemble. Fig. 1 depicts our approach. We use data and

• The authors are with the University of Konstanz, Konstanz 78464, Germany. E-mail: {bruno.schneider, florian.stoffel, keim}@uni-konstanz.de, dominikjaeckle@gmail.com, {diehl, fuchs}@dbvis.inf.uni-konstanz.de.

(Corresponding author: Bruno Schneider.)

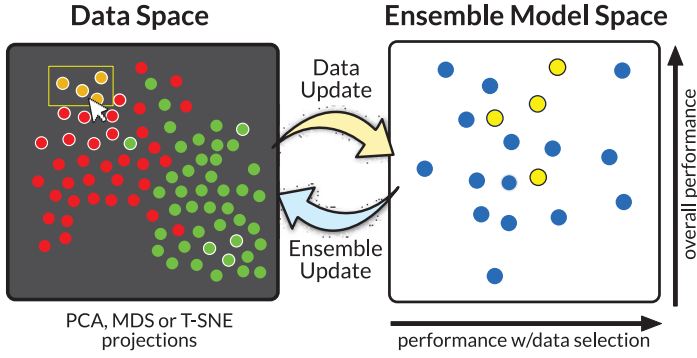


Fig. 1. Visual integration of the data and ensemble model space. Left: The classification results are displayed in scatter plots. The user can decide between a linear (MDS, PCA) or non-linear (t-SNE) projection technique that transforms the results to a two-dimensional scatter plot. A manual selection in the data space triggers a *data selection update*. Right: The model space depicts every single model and allows to compare them by customizing the axes; herein, we contrast the overall performance with the performance w/ data selection. The interactions in the model space trigger an *ensemble update* with immediate impact on the data space.

planar projections to reveal linear (PCA [15], MDS [11]) or non-linear patterns (t-SNE [27]) in the data points or models. Besides, we also offer binned visualizations of the data space to show characteristics of each class that the projections can occlude. The data points are binned per class label and data dimension (or data similarity, see details in Fig. 4). This representation enables the identification in each class of local areas of classification errors and areas of high classification certainty or uncertainty, respectively.

In this work, we claim the following two-fold contribution towards enabling the visual analysis of classification results in Ensemble Learning: First, *the tight visual integration of the data and the model space*. Second, *a workflow that builds upon the visual integration and enables the effective exploration of models and classification outputs*. The visual integration allows to manipulate and explore the impact of each data object and model in a straight-forward manner. Key to our concept is the role of the user, who aims at forming hypotheses and gaining new insight based on the task at hand. Therefore, we relate our contributions to existing theoretical frameworks on human-centered machine learning and provide visual guidance to identify effective models not selected by the automatic search in first place. One can then experiment with alternative ensemble model selections and seek for local improvements based on the constraint that the overall performance is not impaired. The views update on the fly, enabling the user to retrace the impact on the classification outputs. We apply our approach to binary and multiclass classification problems.

Our target users are model developers that can benefit from the explorative capabilities of our approach, as well as domain experts in their classification problem of choice. These experts can express preferences concerning one class or region of the data space, and our tool takes care of finding a proper combination of models to fit these needs.

2 RELATED WORK

Our work builds upon the idea of visually integrating the space of *machine learning* models and the data space, thus enabling the exploration of the impact of each data object and model. Following, we discuss related work from

ensemble learning and interactive model space visualization. Our approach does not aim at retraining the models but at finding effective model combinations that were not given by the automatic search.

2.1 Ensemble Learning

Classifier ensembles aim at combining the strengths of each classification model. To build ensembles, it is necessary to generate a variety of models and then to combine their results. The first step—generating the diversity of models—can be accomplished by making use of different strategies. Several ensemble learning philosophies [17] and methods for combining the classification outputs [31] exist. For example, the same model can be trained successively with different subsets of the data [5], [14], with different types of models [21], [41] (e.g., Decision trees, K-nearest neighbors), or with combinations of strategies such as the mixture of bagging [5] and random combinations of strategies in the Random Forests [6].

In our case, we follow the strategy of producing distinct types of model. With multiple types of classifiers, it is necessary to define which of these types will take part in the ensemble, and this model generation procedure and the multitude of possible combinations motivated the use of data visualization to support this task. Conversely, the model diversity produced by the other strategies is often given by the design of the respective algorithms. In these cases, it is only necessary to set a base classifier, and an automatic process generates all the other models in the background (e.g., the AdaBoost M1 method [14], in which usually Decision Stump trees are the base classifier to produce ensembles using a boosting strategy).

In particular, we worked with Multiple Classifier Systems, in which there is an overproduction phase and the generation of big model libraries (with hundreds or even thousands of models because the analyst typically does not know beforehand which model types will perform well together). Then, with the big model libraries, there are several search algorithms that were developed to look for the best possible combination of models automatically (e.g., GRASP [26], [43], evolutionary algorithms [1]), without experimenting with all the possible combinations due to the complexity of this combinatorial problem. In our work, we use a search selection algorithm developed by Caruana et al. [10]. However, using visualization and interaction we enable the user to update *on-the-fly* the ensemble selection and instantly see the changes in classification outputs. This workflow fits into the *interactive machine learning* concept presented by Amershi et al. in [2], in which the authors refer to the user updates as rapid and focused. Conversely, the fully automated selection method requires to restart the algorithm from the beginning if the user is not satisfied with the results, a time-consuming process.

2.2 Interactive Model Space Visualization

Following, we provide an overview of visualization techniques to represent the model space, also called the *model landscape*. Building upon the well-known visualization methods, we then discuss interactive approaches introduced to steer the performance of classifier ensembles.

Rieck et al. [33] used scatter plots for representing regression models and to perform a comparative analysis of

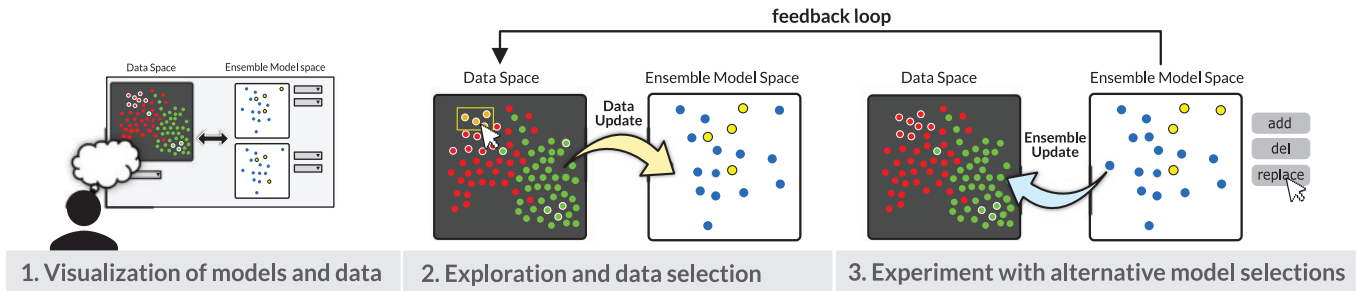


Fig. 2. Our process for exploring ensembles of classifiers starts with the visualization of the classification outputs and the combination of models that produced the corresponding classification (1). Then, it allows the user to explore and select regions of interest in the data space (2), updates the model space to show how each model classifies the current data selection, and allow inclusion, replacement or removal of models from the ensemble (3). At each change in the ensemble configuration, the visualization of the classification outputs is updated accordingly, which introduces a feedback loop that can lead to new rounds of interaction with the system.

competing models. In contrast, Olah [29] also shows groups of models using scatter plots, but for representing distinct Neural Network architectures to classify images of handwritten digits. Similarly, Padua et al. [30] represented collections of *Decision Trees* using several linked visualizations, in which the users can explore large portions of the parameter space of these models and assess the predictive quality of trees derived from several combinations of parameters. In all cases, a positive aspect of building compact visual representations of models is that one can then easily link them to the data, an essential step in better understanding machine learning models [4]. However, it is still missing the extension of these ideas to ensemble learning applications.

The analysis of classifier models through interactive visual interfaces is an active area of research [25]. Talbot et al. [40] present *EnsembleMatrix*, in which the user can interactively build and steer the performance of ensembles of classifiers. In [18], Kapoor et al. also present an interactive tool called *ManiMatrix*, in this case for the improvement of individual classifiers. In both cases, *confusion matrices* appear as a central component. In *ManiMatrix*, the users can express their preferences w.r.t. decision boundaries among classes using a confusion-matrix. In *EnsembleMatrix*, the matrices support the decision of which combination of classifiers works better when building ensembles.

However, despite the compact and efficient information about the class confusion that the matrix-based approaches convey, it is still aggregated data about performance that does not go until the bottom level of the errors with individual data points. To provide this level of access and better visualize where are the errors coming from, we worked with a representation of the data space that shows this level of detail. Also, due to scalability issues, the use of one confusion matrix for every classifier in *EnsembleMatrix* is not applicable to our case, in which we had libraries with hundreds of models for building MCS.

In *EnsembleMatrix*, the ensembles were built from a limited and small number of candidate models, and not in the same way that happens in our context of building MCS. Regarding giving access to the data instead of only showing aggregated information about model errors, Ren et al. [32] pointed recently this limitation of most current systems. They presented a solution for multiclass problems in which they visually compare different models with similar performance but with very distinct behavior w.r.t. to the classes and local regions of the data space. *ModelTracker* [3] also

provided access to the data level for model performance analysis.

We go in the same direction of revealing errors that are not visible when aggregated but we do that in an ensemble learning context. While our interactive visual approach supports the overall improvement of classifier models, we mainly focus on the integration between classification results and models and propose a workflow for effective analysis. With our approach, we give direct access to any model or data point, thus enabling the direct manipulation of these objects and bringing the possibility of locally adjust the ensemble behavior accordingly to the user preferences, when several alternative model selections are possible.

3 INTEGRATING DATA AND CLASSIFICATION MODEL SPACES

We propose a visual analytics approach for the exploration of model and data spaces in ensembles of classifiers. We work with Multiple Classifier Systems (MCS) and introduce a data-guided and user-centered process for interacting with data and models in this context (see Fig. 2). In addition, the direct linkage of data and models is a central component of our workflow, because it allows the user to manipulate objects in any side and see the impacts on the other side instantly, by means of interaction and data visualization.

MCS are often generated from huge model libraries of several types of classifiers with different parameter settings. The process does not depend on previous knowledge about which models perform better for the data and classification task at hand. Several models are produced and an automatic search step looks for the best possible combination of models that deliver higher performance when combined in an ensemble of classifiers. In our workflow, we build a MCS using the standard automatic approach previously described. Then, we initialize our tool, in which we can visualize the models and the classification outputs produced by the initially automatically selected ensemble. Our starting point with our visual analytics approach is after the automatic construction of MCS.

In our tool, we have a visualization panel that represents the classification outputs (the data space, see Fig. 3 (1)), and two linked other ones that show the classification models accordingly to selectable performance and diversity measures (the model space, Fig. 3 (3)). Importantly, we show not only the models that were automatically selected and

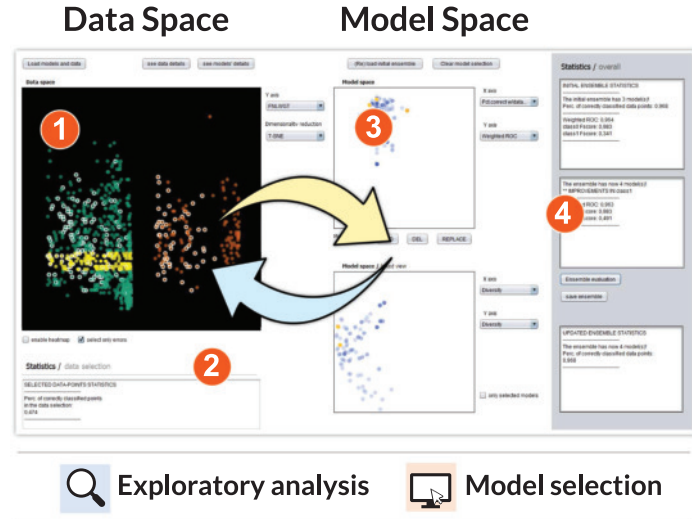


Fig. 3. *Overview of our tool.* The user can select regions of the classification outputs (1), see how the models classified these areas (2), interact with a preloaded collection of models adding, replacing or removing them to update the ensemble (3), and track the performance while interacting with the models (4). Above, we also show the icons we use throughout this paper to identify the two tasks we support: *Exploratory analysis* of data and models spaces, and *Model Selection*.

correspond to the initial ensemble configuration, but also show the whole model library that was used in the beginning of the construction. Therefore, it is possible to add, replace or remove models at any time in the ensemble. With our approach, the process of exploring the model space is driven by the user interest in particular regions of the data space. We present our model space exploration process and its feedback loop with greater detail in the next subsections.

3.1 Representing Models and Data

We aim at enabling the user to directly manipulate each data point and each model in our visualizations. With respect to scalability, we have to consider that the model libraries for building MCS can have hundreds of classifiers. Regarding the data space, we visualize a validation or test dataset with unseen data during the training phase of the models. In any case, models or data, we need a visualization that can accommodate these objects at scale. For this reason, we decided to use scatter plots to visualize both. In the model space, each dot corresponds to a classifier model and the color shows if the model is part of the current selected

ensemble or not. Analogously, in the data space, each dot corresponds to one instance of the dataset, and the color indicates the actual label. Besides, a white outline represents mislabeled points.

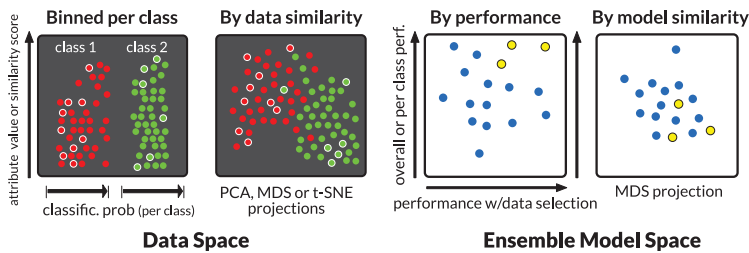
In our tool (Fig. 3), the left-side panel is the data space. The user can decide to start the exploratory data analysis and search for clusters of errors in the classification outputs by selecting to project the data dimensions to the two-dimensional space using data or planar projections.

In addition to the projections, we support the exploration of classification outputs vertically aligned per data similarity or data dimension, and in both cases horizontally aligned by model uncertainty (the *Binned per class* visualization type; see details in Fig. 4). For instance, the user can select one dimension at each time (e.g., *age*) and visualize how the ensemble classifies the data regarding this dimension (see Fig. 5). The user can also decide to see the data points organized by a one-dimensional data similarity measure. The visualization that uses this similarity measure provides the advantage of producing one single plot for the classification outputs instead of having to alternate among plots for each data dimension while keeping the binned per class layout. To obtain the similarity score for each data point given a dataset for classification, we project all data dimensions using *PCA* and take the first component.

Still on the *Binned per class* visualization of the data space, besides using the vertical axis of the scatter plot to show the dimension or similarity score values, we also compute the classification probabilities of the predicted class for each data point and map to the horizontal axis. In addition, we display each class in a different region of the plot and side-by-side, to better distinguish the classification outputs per class (see Fig. 5, in which we have a binary classification problem and the green and orange colors distinguish the data instances from both classes.). However, in scatter plots the overplotting can occur and make it difficult to better identify dense regions in the data. To overcome this problem, we implemented and included a heat map visualization in the data space. At any time, the user can switch between the standard scatter plot and the heat map to show the same data. With this heat map, the clusters with classification errors become more distinguishable.

Concerning the models, we represent them in our tool in the right-side scatter plots (Fig. 3). We precompute measures

Visualization types



Interaction types

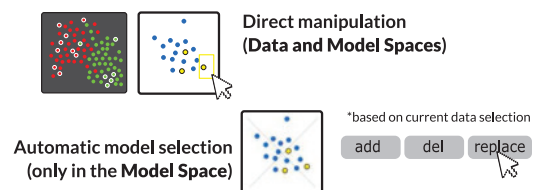


Fig. 4. *Visualization and Interaction types.* To visualize the data space, we provide two alternatives. The first one, *binned per class*, shows the data aligned by measures of model uncertainty and a one-dimensional similarity score or a data attribute value. The second way to visualize the classification outputs is by choosing data or planar projections. The model space also offers two visualization possibilities: the models organized by performance measures or data and planar projections of a model distance matrix. Regarding the interactions, the user can directly select models or data points in the visualization. With the models, there is the additional capability of automatically adding, removing or deleting them based on the current user selection of data, by clicking on the corresponding buttons in the interface.



Exploratory analysis

Hypothesis forming

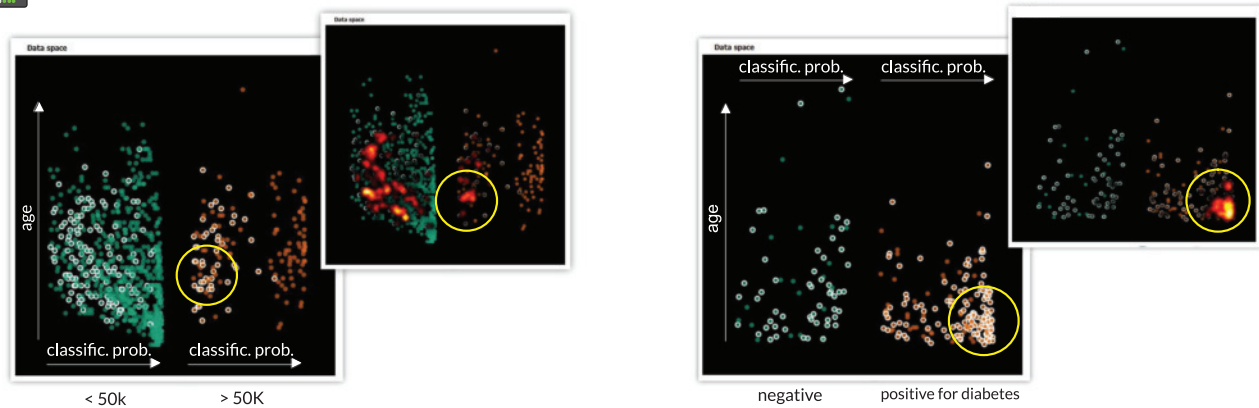


Fig. 5. On the left pair of images, we see the classification outputs for a dataset of individuals that earned less or more than 50K per year. There is a concentration of mislabeled data points in the $>50k$ class (circulated in yellow). In these cases, the ensemble assigned low classification probability, which makes these errors more easy to fix. The smaller image shows the same data with the addition of a heat map. On the right pair of images, we see the classification outputs for a dataset of individuals that tested negative or positive for diabetes. There is a cluster circulated in yellow of mislabeled data points in the *positive* class (dots with a white outline) that corresponds to individuals with lower age. In these cases, the ensemble assigned high classification probability, which makes these errors more difficult to fix.

of performance and diversity for each model and let the user decide which one should be assigned to each axis. We have two linked panels for the models because this layout gives more flexibility to the user. It allows the simultaneous visualization of the model library from different perspectives by assigning different measures to each of the panels. For overall performance, we compute the weighted *Area Under ROC* and weighted *F-Measure* scores. The user can also choose the *F-Measure* score per class. Regarding the model diversity, we use the *Q-statistics* [22], a pair-wise measure that compares the classification of each data point between two classifiers and captures if the models similarly classify the data or not. Then, we build a distance matrix using the *Q-statistics* measure and project this matrix to the two-dimensional space using MDS. In the end, we let the user visualize the outputs of this projection. We can read the model diversity scatter plot in the following way: models more close to each other classify the data in a similar way. Models far away from each other classify regions of the data space differently, despite the fact that they can have similar overall performance.

In the context of ensemble learning, model diversity is an important aspect. Very often, we want to find models that classify distinct regions of the data space not in the same way, because then we can combine the model strengths in a good ensemble. There is research about the role of diversity in ensembles [8] and it is not guaranteed that we can always use this measure to get the best model combination. However, it is still a relevant metric to consider when comparing and visualizing classifiers in ensemble model spaces.

3.2 Interacting with Data and Model Representations

The previously mentioned model-data linkage is a central component of our approach. With this link, we can have the visualization of the data space as an entry point for the user to find regions of interest in the data and the corresponding performance of the models for these regions. This behavior is backed, naturally, by a series of interactions that we implemented in both model and data panel visualizations.

We have, at the end, a process that contains a feedback loop (Fig. 2), in which for any data selection we have corresponding model candidates, and then for any model selection the classification outputs in the data space change again, potentially allowing new rounds of interaction.

When the user interacts with the data space and selects items, we compute the performance of each individual model for the current data selection (percentage of correctly classified data items). Then, the user can decide to use in one axis of the model scatter plots this local performance, and the model space will update accordingly to the selection. In addition, it is also possible to activate a filter that selects only the points that were wrongly classified by the ensemble. This functionality makes it easier to take care of the errors in separate, by facilitating a fast selection of these data instances. The data space always represents the classification outputs (actual class, model uncertainty, and misclassified items) for the current ensemble selection in the model space visualization.

The user can also interact with the models. This can occur in two ways: by direct manipulation, in which the user directly select/deselect models in the scatter plots, based on their position and corresponding metrics assigned to the axes. We also provide the *add*, *del* and *replace* buttons in the interface, which automatically update the ensemble selection accordingly to a previously selected region of interest in the data space (see Fig. 4, *Interaction types*). This functionality facilitates the interaction because if we have several models close to each other in the scatter plots, the direct manipulation is not always a convenient way to precisely pick one of them in particular. To give an example of using the buttons, the user can select a cluster of errors in the data space, and then press the model *replace* button, for instance. This action triggers a method that accesses the performance of all available classifiers in the model space and returns the best and the worst performing model for the currently selected data points. Then, it removes this worst model from the ensemble and includes the best. If the best model for the current data selection is already part of the ensemble, then it

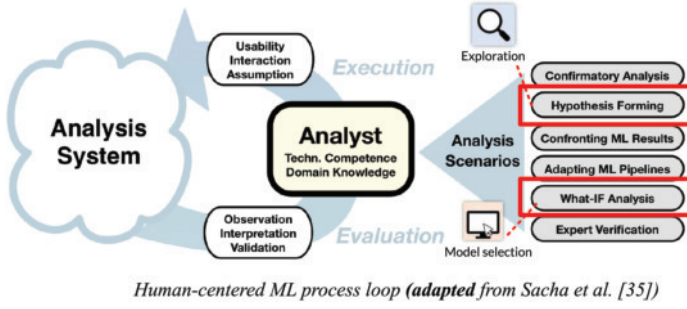




Fig. 6. We show which analysis scenarios we support in a human-centered machine learning framework of reference. Namely, we support the *hypothesis forming* and the *WHAT-IF Analysis* scenarios.

is considered for reinsertion, giving it more weight in this case. We add the constraint that we only pick the best model for the current data selection if it also lies into the last two deciles of overall classification performance. If this condition is not satisfied, our search looks for the next candidate that both performs well locally and globally. Independently of the interaction type of preference, whenever an interaction occurs with the model space and the ensemble changes, we compute the classification outputs for the new model selection and update the corresponding linked panels.

The accuracy of the ensemble with the points in the data space is updated at each interaction with the models and displayed in a text panel with the percentage of correctly classified instances. This computation is very fast to do because we already have precomputed the results for each available classifier in the model space, so it is only necessary to combine the results of the selected models at each time the selection changes. We use the arithmetic mean of probabilities method, which is a standard procedure to combine classification results in an ensemble (for more details and other possible methods, see [23]). To get more significant results about the ensemble performance, we also have a button that the user can press to perform a 5-fold cross-validation evaluation and get the overall and per class ensemble model performance.

Lastly, there is also another filter that makes only the current selected classifiers in the model space accessible, to allow the detailed analysis of the models that are part of the ensemble.

4 THE ROLE OF THE USER IN THE VISUAL EXPLORATION OF CLASSIFICATION MODEL AND DATA SPACES

In this section, we connect our workflow with theoretical frameworks that describe the role of the user in machine learning pipelines. We identify two main tasks supported by our approach: the *exploratory analysis*  of model and data spaces (*hypotheses forming*) and the *ensemble model selection* , in this case by experimenting with alternative ensemble configurations guided by the interplay between those spaces and how the data reacts to model changes (*WHAT-IF analysis*).

In [35], Sacha et al. use the expression *Human-Centered Machine Learning* to present a conceptual framework that describes human interactions with machine learning (ML) components. The authors focus on the combination of ML methods with human feedback through interactive


visualization. The proposed conceptual framework fits any ML method besides classification and describes the role of the analyst in any step of a complete visual analytics/ML pipeline. Other authors also highlight opportunities to combine ML algorithms with human expert knowledge through interactive graphical interfaces. In [2], Amershi et al. discuss the role of humans in *Interactive Machine Learning*, in comparison with traditional machine learning workflows. Still in [35], the role of the analyst in a human-centered ML process loop is shown in more details and considers six analytic scenarios: *Confirmatory Analysis*, *Hypothesis Forming*, *Confronting ML Results*, *Adapting ML Pipelines*, *What-IF Analysis* and *Expert Verification*.

Two of the mentioned scenarios are within the scope of our work, namely the *Hypothesis Forming* and *What-IF Analysis*. The first one corresponds to the *Exploration* task in our categorization. The second, *What-IF Analysis*, corresponds to our *Model selection* task (see Fig. 6).

4.1 Exploratory Analysis

Our workflow suggests that one can explore either the model space, the data space, or both spaces combined. This brings flexibility to the user, who can start the exploration by interacting with the data or classifier models. The exploratory analysis capabilities of our approach go hand in hand with the proposed scenario *Hypothesis Forming* introduced by Sacha et al. [35]. The main idea is to form new hypotheses without having specific knowledge about the data or the models. A means to form new hypotheses is to seek for patterns that reveal trends, clusters, outliers, or any other structure of interest. Such structures help to generate insight and provide helpful information towards understanding data and models. Following, we outline all three scenarios: the model space exploration, the data space exploration, and the combined exploration.

4.1.1 Model Space Exploration

The user *explores*  the model space by assigning different precomputed measures to the scatter plot axes. Going one step further, one can identify clusters of models that perform similarly by deciding to investigate their dissimilarities at a glance. To do so, the user can project the models in a two-dimensional space using MDS. In this particular case, MDS is the rational choice, because it is a linear projection technique that preserves the distances and provides a global view of the models. MDS is applied to the model distance matrix, which we derive using the pair-wise *Q-statistics* model diversity measure. This measure captures differences in the way classifiers label the data points. This way, the user can inspect the impact of the model diversity concerning a particular ensemble selection (see Figs. 7 and 8). If all the ensemble selected models are clustered in the scatter plot representation, this indicates that this ensemble does not have a diverse set of classifiers, and performance played a major role. Such information can support the decision to not spend additional time training alternative model types that can result in new diversity score ranges. We applied our visual approach to analyze model spaces ranging from 100 to 1000 trained classifiers of different types and distinct parameter settings. One can combine these models in several ways to build ensembles.

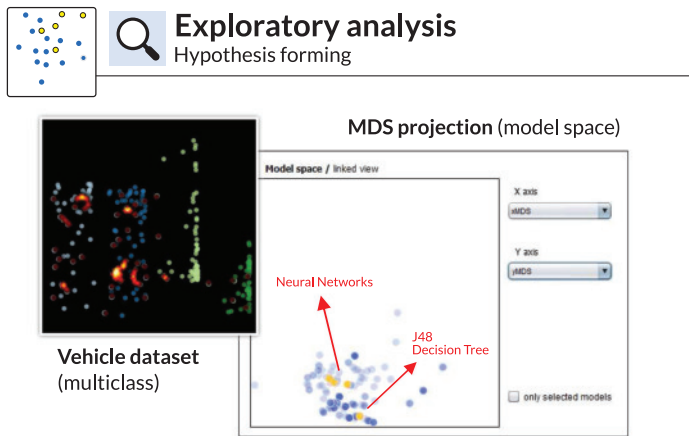



Fig. 7. *Model diversity in a multiclass problem.* The right scatter plot shows an MDS projection of the models automatically selected to classify the *Vehicle* dataset in ensembles of classifiers, marked in yellow. In this case, we run the ensemble automatic selection [10] using a backward and forward search strategy. Even though the neural networks have the best overall accuracy individually and outperform decision trees, a J48 Decision Tree was also selected because in combination with the others it improves the classification globally. In our tool, the individual model performance is accessible in auxiliary text panels and linked visualizations.

4.1.2 Data Space Exploration

In the second scenario, and analogous to model space exploration, the user can decide to *explore*  the data space and corresponding outputs of the current ensemble selection as depicted in Fig. 5. The analysis of the classification outputs with the *binned per class* visualization type (see Fig. 4) reveals the distribution of classification errors. At any point in time, the user can change the visual alignment of the data instances by selecting either a data dimension or the precomputed data similarity measure. While the data dimension supports the identification of classification errors that occur in a particular value range, the data similarity measure reveals errors that can be considered as similar based on the first component of the PCA. We choose PCA over other projection techniques because it is a linear technique, which captures global patterns of the data based on the pair-wise co-variances (a measure of the joint variability). It also tends to retain in the first component more information about the data variance than other methods. This choice provides us with the necessary means to identify similar data points, as well as outliers on a global scale, which were misclassified. Digging into the characteristics of similarly misclassified data points can also give an idea of why they are mislabeled. For example, a single model could be responsible for the misclassification of a point cluster, which is not visible in another view. Note that the application of a planar projection technique has no impact on the classification results at all. The projection does not determine the classification results but facilitates to draw conclusions based on the data characteristics. In both cases, classification errors next to the decision boundaries are typically easier to correct by experimenting with alternative ensemble selections compared with errors where the model assigned a high probability to the predicted class. To change the classification of mislabeled data instances with a high probability, more significant changes in the ensemble selection are necessary. These changes typically propagate across all classification outputs.

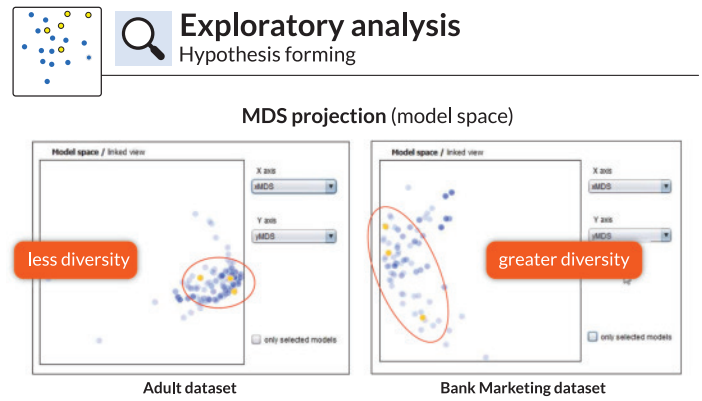




Fig. 8. *Model diversity in binary classification problems.* In the above examples, we use two binary classification datasets, the *Adult* and the *Bank Marketing* datasets. Both scatter plots show MDS projections of models automatically selected to classify each dataset in ensembles of classifiers, marked in yellow. In comparison, the right case shows greater model diversity, represented by the greater distance among models in the projection. The *Bank Marketing* dataset is highly unbalanced, and most of the models with better overall performance show poor performance with the smaller class. Then, a very diverse model appears in the automatic ensemble selection because of its excellent performance with the class with fewer instances, while not hurting the ensemble overall performance.

Besides, the user can also visualize the data and find clusters of errors by exploring two-dimensional data or planar projections. This alternative has the advantage of using both the vertical and horizontal axes of the scatter plots to represent data similarity, thus preserving more information about how similar or dissimilar are the data. In the end, we tackle a combinatorial problem with multiple candidate solutions, which are combinations of models in ensembles of classifiers. Therefore, we offer distinct views on the data, giving more possibilities to the user in identifying regions of interest in the data space that lead to that alternative model combinations.

4.1.3 Exploration of the Interplay Between Model and Data Space

Besides the possibility of exploring both, models and data separately, the full functionality of our approach comes to light when exploring both spaces combined (see the *feedback loop* in Fig. 2). The user can *explore*  the reaction of the data space to the model space, and vice versa. For example, the user can investigate how the ensemble selected models perform with the most prominent clusters of errors in the data space. By choosing a region of interest in the data space, the user can adjust the model space to show the performance of all models regarding the data selection. The user can then focus on the ensemble selected models and see if the models classify clusters of errors similarly or if there are significant differences between models. This information is crucial to get an initial idea of the reason for the misclassification. Either all the ensemble models' perform poorly with the data selection or only a subset of them. In the latter case, it is more likely that an alternative ensemble selection help to fix mislabeled items in the data space.

4.2 Model Selection in Ensemble Learning

Apart from the *Exploration* task, our workflow also supports *Model selection* . This capability brings power to our tool because it enables the user to construct ensemble models

interactively. The user experiments with alternative ensemble configurations and introduces changes in the ensemble model selection, which affect the respective classification outputs. The experimentation typically occurs right after a previous exploration phase. However, here the user ends up with a different ensemble of classifiers compared with the state of the system at the beginning.

We correlate the model selection capabilities of our approach to the *What-IF Analysis* scenario described in [35]. Applying this type of exploration, the analyst can interact with the ML pipeline and observe its reaction to changes. According to our workflow for model selection in ensemble learning, the changes correspond to the modifications that are introduced by the user in the selection of classifiers in the model space.

As mentioned in Section 2.1, the automatic ensemble selection procedure is a huge combinatorial problem and, therefore, does not experiment with all possible model combinations. On that score, it is natural to expect alternative ensemble selections to be in favor of a given class and also preserve the overall classification performance at the same time. In a multiclass problem, the user has a greater chance to improve the performance of the model for one or another class than for all classes together.

The user notion of importance helps to prioritize and decide on trade-offs. Often, it is not possible to improve all regions of the data space at the same time. The chance of identifying clusters of errors through several visualization types, select and inspect the raw data in linked text tables and experiment with alternative ensemble model combinations with *on-the-fly* feedback about model performance is central in our approach. Thus, the user has comprehensive information at hand to decide which region of the data space should be prioritized, taking into account that frequently there is no chance to fix all points together. We refer to this process of prioritizing regions in the data space, as well as looking for alternative ensemble selections, as a task of setting up constraints in the data. Constraints, thereby, refer to areas that the user wants to protect from poor classification performance. A full example and walk-through of this type of exploration appear in the following section.

5 VISUAL ANALYSIS OF CLASSIFICATION RESULTS IN BINARY AND MULTICLASS PROBLEMS

In this section we showcase the full capabilities of our proposed workflow by building Multiple Classifier Systems (MCS) for binary and multiclass application problems. We use state of the art benchmark datasets from the UCI machine learning repository [24], and a collection of existing classification models [16] to build MCS for binary and multiclass problems. For the binary classification problem, we work with two set-ups: a model library with 1,045 classifiers, and a stratified sample with 200 models, for fast training. In both cases, we have 13 different model types and varying parameter settings. With the multiclass experiment, we work with 200 classifiers sampled from a library with 986 models. This library has eleven different model types with varying parameter settings.

We use the ensemble selection algorithm of Caruana et al. [10] to perform initial automatic ensemble selections.

At the beginning of each experiment, our tool initializes with the classifier libraries, the automatic ensemble selection performed by Caruana's algorithm, and the initial classification outputs. Users can freely experiment with alternative ensemble selections in the model space, and see how the data space reacts to these changes.

In Section 5.1, we group and describe all we have done to prevent overfitting, a common issue that comes with the usage of ensembles. In Sections 5.2 and 5.3, we describe two experiments following the workflow presented in Fig. 2, contemplating the interactions with models and data described in Section 4.

5.1 Overfitting and Model Generalization


Ensembles can overcome the performance of individual classifiers, but the benefit comes with the higher risk of overfitting [9]. This problem happens, for instance, when a model captures some peculiarities of the training data, such as those caused by noise in collecting the learning examples [44]. Then, these are wrongly recognized by the learner as the underlying truth. An ensemble can be complex enough to fit the training data perfectly, but too much model complexity leads to a poor generalization with new data.

To prevent the problem mentioned above, the main functionality we introduce is the chance of replacing existing models in an ensemble. By keeping it compact, we avoid excessive model complexity, which would have been the case if we employ only an additive approach. However, besides preventing the overfitting and aiming to keep an optimum global model performance with new data, we allow the user to favor one particular class as Kapoor et al. did in [18]. In this work, the authors refer to existing classification problems with distinct mislabeling costs.

Regarding the evaluation of the ensemble selection obtained after user interaction, we use two methods to assess model generalization. In the binary classification problem (Section 5.2), we split the data into train, validate, and test sets. The first split we use to train the individual classifiers. The second, validate data, is the one the user interacts with to adjust the ensemble and get instant updates on the model performance metrics. Then, we compare the performance of the automatically selected ensemble and the user-adjusted ensemble using a held-out test dataset and report the results in the following subsection. With the multiclass problem (Section 5.3), we use only the train/test data splits because the dataset is not big enough for more splits. The user interacts with the test data and modifies the ensemble. Then, we update the performance metrics using a 5-fold cross-validation evaluation on all the data.

Finally, we use bias/variance decomposition and report these measures to collect additional evidence that the user does not go in the overfitting direction after adjusting the ensemble selection. This method also allows inspecting if the efforts to avoid overfitting (variance) do not fall into the opposite error of underfitting (bias) [13].

5.2 Ensemble Model Selection in a Binary Classification Problem

In the following experiments, we showcase how we support ensemble *model selection*  in a binary classification problem, using model libraries of 200 and 1045 classifiers.

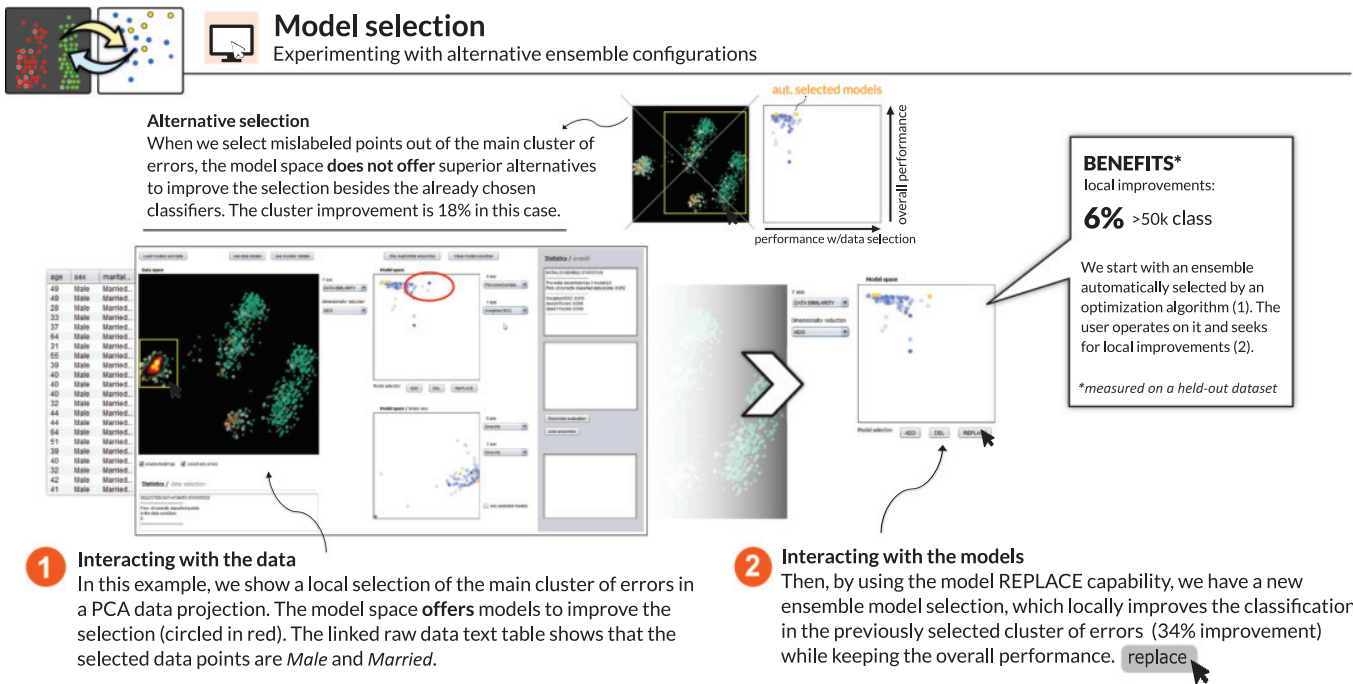


Fig. 9. *Binary classification with the Adult dataset*. The user interacts with the data and models to find a model selection that improves the >50k class without hurting the overall performance. The automatically selected ensemble (1) has a 0.91 overall performance ROC score, a 0.90 F-Measure score for the <50k class, and a 0.67 F-Measure for the >50k class. After user interaction (2), the corresponding scores with the new ensemble selection are 0.91, 0.9 (0.897) and **0.71**.

5.2.1 Adult Dataset with a 200-Models Library

We use in this example the binary classification *Adult* dataset, in which the task is to predict whether the income of an individual exceeds 50k US dollars per year based on census data. We work with train, validate, and test (held-out) datasets, with 5000, 2500 and 2500 instances, respectively. To assess model generalization, we use the test (held-out) dataset to compute the performance metrics of the model obtained after user interaction with the validate set.

We start with a model library of 200 classifiers, and the automatic selection procedure gives an ensemble with three models at the beginning (a Bayes Net, a Decision Stump base model using AdaBoost and a REPTree base model with bagging). We experiment with different visualizations of the classification outputs, and the PCA projection enables us to identify clusters of errors in the data space easily. We try selecting the most prominent cluster and improve the classification in this region (see Fig. 9(1)). After selecting the biggest cluster, we adjust the model space to show the best models for the current data selection. Next, we press the model *replace* button once (Fig. 9(2)). This interaction removes the worst model for the current data selection and searches for the best one for inclusion. In this case, the Bayes Net was replaced by another model of the same type with different parameters settings, available in the model library but not automatically selected at the beginning. These changes improve the classification of the >50k class by six percent while keeping the overall performance (0.91 using the ROC score).

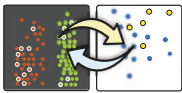
The *Adult* dataset is unbalanced, with more instances in the first class (<50k/year). Most of the classifiers in the model library perform better with this class, which gives room for improvement with the other one (>50k/year). To assess how well the new ensemble selection generalizes after user interaction, we estimate the bias and variance for the new ensemble selection (see the table in Fig. 12). The

bias is the same, which shows that the new model selection is as accurate as the initial one. The variance almost does not change, which gives us the information that the new selection potentially generalizes well to unseen data and overfitting is not a major risk.

5.2.2 Adult Dataset with a 1045-Models Library

In the second example with the *Adult* dataset, we use a different model setup. In this case, we use a model library with 1045 classifiers, instead of 200. The automatic ensemble selection picks seven decision tree models from the initial collection: two base-classifier trees with AdaBoostM1, three with bagging and two standard J48 decision tree models. Regarding the preparation of the datasets, we use the same setup of the previous experiment (train, validate, and test (held-out) datasets, with 5000, 2500 and 2500 instances, respectively).

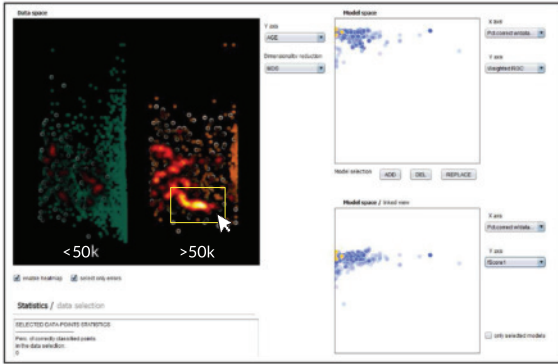
We start the interactive exploration of the data space by selecting the *Age* data dimension on the vertical axis of the data space plot with the aim of finding clusters of mislabeled items by this dimension. In the visualization of the data (see Fig. 10(1)), we have the two classes positioned horizontally in opposite areas of the scatter plot. The classification probability, indicated by the horizontal position of each data point, allows the analysis of the error distribution per class. So far, we identified areas that are likely to contain more errors in the classification than others. Although, the visualization suffers from overplotting, which hinders us from perceiving the actual data point density in the corresponding regions of the scatter plot. To overcome this limitation, we enriched the point-based scatter plot visualization with an inverse distance weighting-based point density visualization [38]. The actual density of data points can be perceived efficiently using the resulting heat map. In areas with overplotting caused by clusters of mislabeled data, the heat map helps to



Model selection

Experimenting with alternative ensemble configurations

VALIDATION DATA



1 Local selection of mislabeled data points guided by the clusters in the heatmap visualization



2

Experiment with alternative ensemble configurations, by replacing classifiers in the initial ensemble with models that best classify the data selection and have good overall performance



3

After replacing models, the visualization shows the improvement in the user initially selected region

replace

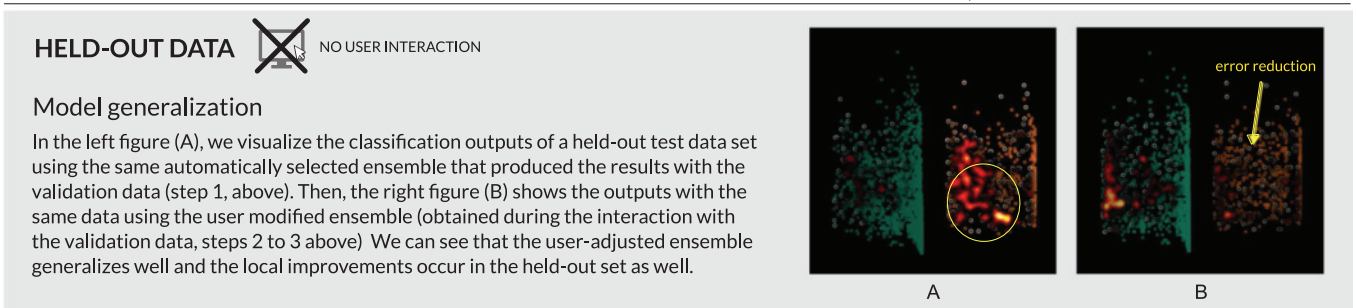


Fig. 10. Visualizing clusters of errors and searching for alternative ensemble configurations based on user-defined regions of interest in the data space. The automatically selected ensemble (1) has a 0.91 overall performance ROC score, a 0.91 F-Measure score for the $<50k$ class, and a 0.67 F-Measure for the $>50k$ class. After user interaction (2), the corresponding scores with the new ensemble selection are 0.91, 0.90 and **0.72** (7 percent class improvement). The improvement in the selected cluster of errors is 21 percent in this case. Besides reporting the performances, we visualize how the interactively chosen ensemble performs with unseen data (in *HELD-OUT DATA*, A and B).

distinguish significant variations in the quantity of these errors.

Then, we choose the class that worsens the ensemble performance ($>50k$ class, Fig. 10(1)) and select the densest regions using the heatmap to identify areas with clusters of mislabeled data points. To do so, we adjust the model space scatter plots to show both the performance of the ensemble with the current data selection and the overall performance. In the top-right area of the models scatter plot, the region that shows classifiers with good local and overall performance, we discover models that were not included by the automatic selection procedure (Fig. 10(2)). The automatic ensemble selection [10] does not take all possible combinations of models into account. In consequence, it is always possible to find competitive new ensemble configurations.

We start the interaction with the models by replacing the worst performing model for the current data selection. Next, we repeat this procedure by pressing the *replace* button again, backed by the performance statistics panel and the data space visualization updates. Last, we make the ensemble even more compact to reduce the chances of overfitting and press the *del* button once. This action removes from the current ensemble the worst model for the initially selected data points. We see in the data space that we reduced the errors in the $>50k$ class significantly, with the cost of increasing the errors in the $<50k$ class by a fraction (see details in Fig. 10). The final ensemble selection contains

a new Bayesian Network model inserted twice, together with decision trees we had at the beginning.

5.3 Ensemble Model Selection in a Multiclass Problem

In this example, we showcase how we support ensemble *model selection* in a multiclass problem using the *Vehicle* dataset. The classification task is to label a given silhouette as one of four types of vehicle, using a set of features extracted from the silhouette. We divide the data into 564 training and 282 test instances. The user interacts with the test data. When the model selection changes, we update *on-the-fly* the performance statistics using 5-fold cross-validation on this data. We initialize our tool with an automatically selected ensemble with three neural networks models. We use the F-Measure score throughout this experiment, both to compute the weighted overall and per class performances.

We start the exploration of the data space by trying to improve the classification in the region with most of the points of the worse performing classes (see Fig. 11). The visualizations that use dimensionality reduction allow to inspect the mentioned region. We select among them the t-SNE projection because it provides the best class separation in this case. Next, we select the points in this region with most worse classified instances (*opel* and *saab* classes). We use the linked data text panel to confirm the data labels. We try replacing the model with the worst performance for the

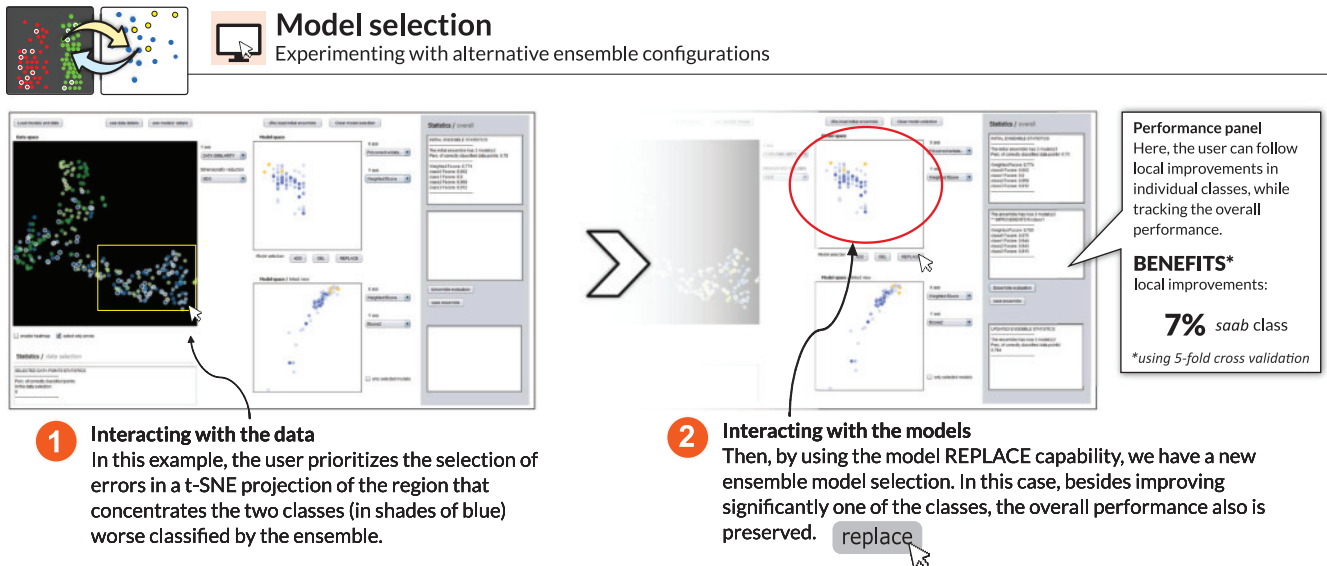


Fig. 11. *Multiclass classification*. In this example with the four-classes *Vehicle* dataset, we describe above how the user interacts with the data and models (1) to find a better ensemble selection that improves one class without hurting the overall performance (2). We start with a model selection produced by an automatic search algorithm and explore alternative combinations using our tool.

current data selection with the best one. We press once the button *replace* in the model space for that. Automatically, our tool removes one of the neural networks from the ensemble selection and insert one learner based on a multinomial logistic function. We still have three models after the interaction, but not all of them are neural networks anymore.

We evaluate the performance of the new ensemble selection by performing cross-validation and verify it using the statistics panel in our tool. The results are satisfactory, because one of the classes show now a significantly better classification accuracy, with 7 percent improvement (class

saab). Meanwhile, the overall performance is still as good as the initial one, showing even a small increase (0.77 initially, and 0.78 after user interaction).

The *Vehicle* dataset we use in this example is balanced, with almost the same number of instances for all the classes. However, the performance of the initial automatically selected ensemble is way better with two of these classes (*bus* and *van*, 0.96 and 0.91 respectively) than the other ones (*opel* and *saab*, 0.66 and 0.60 respectively). So, there was room for improvement in the worst classified classes. We do that by experimenting with alternative configurations using our approach, and we achieve this goal without hurting the overall performance. After user interaction, the new ensemble selection shows a performance of 0.94 and 0.91 with the *bus* and *van* classes. With the *opel* and *saab*, the new performance is 0.67 and 0.64, respectively. We observe that the benefit of improving the *saab* class comes with a small loss in performance in the *bus* class.

Very importantly, we estimate the bias and variance of the initial ensemble (before interaction) and the final model selection obtained after the user interaction (see table in Fig. 12). The bias is almost the same, which confirms that the new ensemble keeps the overall predictive performance of the automatic model selection. Additionally, the variance is the same, which tells us that the new ensemble has the same capacity to deal with variations in unseen data distributions than the initial one. The fact that we use the *replace* feature instead of an additive approach is crucial to avoid data overfitting while bringing clear benefits to regions of the data space initially worse classified.

6 DISCUSSION AND FUTURE WORK

In this section, we emphasize the main strengths of our work and indicate directions for future research on the integration of classification model and data spaces.

Data versus Feature-Space: Many related works that use visualization for the inspection, attribute selection (also called *feature selection*), or in general the improvement of a part of the classification problem exist [7], [20], [28]. Those

Ensemble model generalization analysis



BEFORE INTERACTION

Datasets	ensemble size	bias	variance
Adult	3 models	0.13	0.03
Vehicle	3 models	0.18	0.15
Adult (w/ a 1045-models library)	7 models	0.12	0.07



AFTER USER INTERACTION

Datasets	ensemble size	bias	variance
Adult	3 models	0.13	0.04
Vehicle	3 models	0.17	0.15
Adult (w/ a 1045-models library)	6 models	0.13	0.08

Fig. 12. We estimate the model bias and variance before and after user interaction, with different data sets and corresponding experiments. When we compare in each case (dataset) the values before and after interaction, they almost do not change. This decomposition of the error using both measures shows that the new ensembles obtained by the user do not move substantially in the direction of overfitting (variance) nor underfitting (bias).

techniques stay at attribute level and allow the exploration of the attribute space with respect of the importance or the added value of an attribute. Typically, the goal is to adjust the attribute set, and in consequence the re-training of classification models, which is fundamentally different to the model selection task we are aiming at. We provide the complete data space in two-dimensional scatter plots, as illustrated in Fig. 1, with focus on the exploration of classification errors, as these are the starting point for further adoptions of the ensemble. Compared to existing work, this effectively reduces the abstraction between the data input and the classification problem, as we omit the attribute extraction and the corresponding data transformation. Instead, we allow the user to directly work with the data that is subject to the classification, which fosters reasoning and enables findings on data record level, which is the key feature of our work.

Interactive Exploration: A core part of our contribution is the manual selection of regions of the data space, which is a task that can be automated, for example, by utilizing an interestingness measure. However, the search space enumeration is a very costly operation, in terms of computing time and the required computing power, as there are many different sets of data points to enumerate. To overcome this problem, we present the user scatter plot visualizations, where visual patterns created by the point positions as well as their visual mapping, guide the user to interesting areas. Additionally, the user can bring in expert or domain knowledge about the data to make informed guesses of interesting local regions as a starting point for further examination and exploration. When it comes to interactive model selection, i.e., the adaption of the classifier ensemble, we protect the performance of the ensemble by providing a linked text panel that shows if the global performance does not get worse when the model selection changes. By doing that, we do not require the user to understand all model differences at all. Still, the user can decide to add, remove or replace existing models in the ensemble, in correspondence with the selection of a region in the data space. In the current implementation, when the user presses the model *add* or *replace* button more than once, our algorithm favors the reinsertion of the same best model for the current data selection. Future work could extend this mechanism to support optimization methods that perform a neighbor search, and offer alternatives to reinsertion.

Generalization: Our approach is clearly suited for wider application beyond the use that we illustrated in Section 5. Formally, our proposed workflow is comprised of the classification problem, the input data and a collection of classifiers. The classification problem can be a binary or multiclass problem, which makes the workflow applicable to all kinds of classification problems. The collection of classifiers is not restricted to Multiple Classifier Systems. Random forests, or more general, any other hybrid information system, is also suitable for our approach. Additionally, we also support varying parameter spaces, varying model families and arbitrary combinations of them. In consequence, our workflow is not only suitable for the selection of model families, as we demonstrated in this paper, but also for parameter space exploration. An issue of classic feature-based visualizations of classification problems is scalability. Visually, we already introduced density-based heat maps

as a counter-measure to be able to scale to large data sets or model spaces. Therefore, limitations are imposed only by the available computing power, and in consequence the ability to support interactions in the model and data space.

Visualization: As described in Section 3.1, we visualize the data and model spaces using scatter plot visualizations, where each point represents a data record, or a classification model from the classifier model library, respectively. To get an overview of the classification outputs, the user can generate the two-dimensional data space scatter plots by applying state of the art dimensionality reduction, namely projection techniques such as PCA, MDS, or t-SNE. The choice of the projection technique depends on the data characteristics the user aims to consider during analysis. Therefore, we have to differentiate between linear (PCA, MDS) and non-linear (t-SNE) projection techniques. While linear techniques provide a global view on dissimilarities, non-linear techniques look at local characteristics. Then, using a linear technique always provides an overview of how all data records are connected to each other. The visual proximity between data records has a meaning, which is defined by the similarity measure. For example, MDS is based on the pair-wise distances and PCA is based on the pair-wise co-variances. In contrast, non-linear techniques, such as t-SNE, look at local dependencies, where visual distances have no specific meaning other than a separation of dissimilar data records.

We also provide *binned per class* representations as depicted, for example, in Fig. 10. We designed our data exploration process taking into account the possibility of filtering the data space by one dimension at each time, which potentially allows the user to do meaningful selections and identify particular regions of major interest. This enables the user to deeply explore the classification outputs, understand their relationship with the selected data dimension and identify clusters of errors that could not be distinguishable only with overview visualizations (like the ones obtained with the data and planar projections). The classification results binned per class provide an additional way of finding local patterns. The bins for each class have limitations regarding their scalability, yet support a wide range of the existing classification problems (see the survey presented in [32]).

Scatter plots are prone for overplotting, which could result in potentially wrong impressions of the data distribution, as it is nearly impossible to perceive the number of overplotted data points correctly. To cope with this issue, we integrated a heat map overlay, which displays the density of data points using a continuous interpolation, mapping point density to colors ranging from black over red and yellow to white. Alternatives to this approach, such as scatter plot matrices (SPLOMs), are available, although, they do not support the idea of an integral data space visualization. Instead, they display pairwise attribute combinations, which are subsets of the data space. Similarly, small multiples or glyph-like settings are possible, but still, it has to be decided what information is shown by the visualization, as well as how to order them meaningfully. Because the two-dimensional position of the points in the scatter plots indicates their position in the data space, we use in the heat map the color of the data points to indicate errors, as we are especially interested in data points that are classified wrongly. Future work is necessary to assess how well our

data and models visual integrative approach fits other visualization techniques.

Overfitting: In our experiments, we noticed that adding many models to the initial automatic ensemble selection can increase the performance with a validate data subset, but the new model selection often does not generalize well, and model overfitting is a problem. In these cases, we got substantial differences in the variance values, when comparing the distinct ensemble models *before* and *after* interaction, for the same dataset. So, we recommend using the model *replace* feature, which allows local improvements while adequately generalizing with unseen data. For future work, we plan to integrate additional visualizations that facilitate the analysis of how well the model obtained through interaction performs with new and unseen data, e.g., the *bull's eye* diagram to visualize model bias and variance [13].

Evaluation: We performed a quantitative evaluation of our methods. We measure the gain in performance obtained through local classification improvement of regions of the data space. We also assess how well the user-adjusted ensemble generalizes to new data. However, research on the broader impacts of giving more roles to the human in the model building process is still an open field [42]. We plan to extend the scope of the evaluation we have done so far, and perform controlled user studies to assess both qualitative and quantitative aspects related to human participation in the construction of classifiers.

7 CONCLUSION

We foster the use of visual methods for exploring model and data spaces, thus enabling the experimentation with alternative models selection in ensemble learning. Our integrative approach enables a feedback loop that keeps the user always in control of any model selection change introduced in ensembles of classifiers. We use Multiple Classifier Systems to instantiate our ideas and explore those abstract spaces. However, we can generalize and extend our workflow to any type of classifier models, combined in ensembles or not, what gives plenty of opportunities for visualization research on correlated topics.

ACKNOWLEDGMENTS

This paper has been supported by the German Research Foundation DFG within the project *Knowledge Generation in Visual Analytics* (FP 566/17), and by the CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brazil. D. Jäckle is currently an independent researcher.

REFERENCES

- [1] L. M. Almeida and P. S. Galvão, "Ensembles with clustering-and-selection model using evolutionary algorithms," in *Proc. 5th Brazilian Conf. Intell. Syst.*, 2016, pp. 444–449.
- [2] S. Amershi, M. Cakmak, W. B. Knox, and T. Kulesza, "Power to the people: The role of humans in interactive machine learning," *AI Mag.*, vol. 35, no. 4, pp. 105–120, 2014.
- [3] S. Amershi, M. Chickering, S. M. Drucker, B. Lee, P. Simard, and J. Suh, "Modeltracker: Redesigning performance analysis tools for machine learning," in *Proc. 33rd Annu. ACM Conf. Human Factors Comput. Syst.*, 2015, pp. 337–346.
- [4] E. Bertini and D. Lalanne, "Surveying the complementary role of automatic data analysis and visualization in knowledge discovery," in *Proc. ACM SIGKDD Workshop Visual Analytics Knowl. Discovery: Integrating Automated Anal. Interactive Exploration*, 2009, pp. 12–20.
- [5] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.
- [6] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
- [7] M. Brooks, S. Amershi, B. Lee, S. M. Drucker, A. Kapoor, and P. Simard, "Featureinsight: Visual support for error-driven feature ideation in text classification," in *Proc. IEEE Conf. Visual Analytics Sci. Technol.*, 2015, pp. 105–112.
- [8] G. Brown and L. I. Kuncheva, "good and bad diversity in majority vote ensembles," in *Proc. Int. Workshop Multiple Classifier Syst.*, 2010, pp. 124–133.
- [9] R. Caruana, A. Munson, and A. Niculescu-Mizil, "Getting the most out of ensemble selection," in *Proc. 6th Int. Conf. Data Mining*, 2006, pp. 828–833.
- [10] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes, "Ensemble selection from libraries of models," in *Proc. 21st Int. Conf. Mach. Learn.*, 2004, Art. no. 18.
- [11] T. Cox and A. Cox, *Multidimensional Scaling, Second Edition*, London, U.K./Boca Raton, FL, USA: Chapman & Hall/CRC Press, 2000.
- [12] T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. Int. Workshop Multiple Classifier Syst.*, 2000, pp. 1–15.
- [13] P. Domingos, "A few useful things to know about machine learning," *Commun. ACM*, vol. 55, no. 10, pp. 78–87, pp. 2012.
- [14] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. 13th Int. Conf. Mach. Learn.*, 1996, pp. 148–156.
- [15] I. Jolliffe, *Principal Component Analysis*. Berlin, Germany: Springer, 1986.
- [16] R. Jung, "ensemblelibrary Version 1.0.6," 2017. [Online]. Available: <http://weka.sourceforge.net/doc.packages/ensembleLibrary>
- [17] A. Jurek, Y. Bi, S. Wu, and C. Nugent, "A survey of commonly used ensemble-based classification techniques," *Knowl. Eng. Rev.*, vol. 29, no. 5, pp. 551–581, 2014.
- [18] A. Kapoor, B. Lee, D. Tan, and E. Horvitz, "Interactive optimization for steering machine classification," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 2010, pp. 1343–1352.
- [19] R. Kohavi and D. H. Wolpert, "Bias plus variance decomposition for zero-one loss functions," in *Proc. 13th Int. Conf. Mach. Learn.*, 1996, pp. 275–283.
- [20] J. Krause, A. Perer, and E. Bertini, "Infuse: Interactive feature selection for predictive modeling of high dimensional data," *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 12, pp. 1614–1623, Dec. 2014.
- [21] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*. Hoboken, NJ, USA: Wiley, 2004.
- [22] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Mach. Learn.*, vol. 51, no. 2, pp. 181–207, 2003.
- [23] L. I. Kuncheva, C. J. Whitaker, C. A. Shipp, and R. P. Duin, "Limits on the majority vote accuracy in classifier fusion," *Pattern Anal. Appl.*, vol. 6, no. 1, pp. 22–31, 2003.
- [24] A. Frank and A. Asuncion, "UCI machine learning repository. Irvine, CA: University of California, School Inf. Comput. Sci., vol. 213, p. 2, 2010.
- [25] S. Liu, X. Wang, M. Liu, and J. Zhu, "Towards better analysis of machine learning models: A visual analytics perspective," *Visual Informat.*, vol. 1, no. 1, pp. 48–56, 2017.
- [26] Z. Liu, Q. Dai, and N. Liu, "Ensemble selection by grasp," *Appl. Intell.*, vol. 41, no. 1, pp. 128–144, 2014.
- [27] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.
- [28] T. May, A. Bannach, J. Davey, T. Ruppert, and J. Kohlhammer, "Guiding feature subset selection with an interactive visualization," in *Proc. IEEE Conf. Visual Analytics Sci. Technol.*, 2011, pp. 111–120.
- [29] C. Olah, "Visualizing representations: Deep learning and human beings," 2015. [Online]. Available: <http://colah.github.io/posts/2015-01-Visualizing-Representations/>
- [30] L. Padua, H. Schulze, K. Matković, and C. Delrieux, "Interactive exploration of parameter space in data mining: Comprehending the predictive quality of large decision tree collections," *Comput. Graph.*, vol. 41, pp. 99–113, 2014.
- [31] M. P. Ponti Jr, "Combining classifiers: From the creation of ensembles to the decision fusion," in *Proc. 24th SIBGRAPI Conf. Graph. Patterns Images Tutorials*, 2011, pp. 1–10.
- [32] D. Ren, S. Amershi, B. Lee, J. Suh, and J. D. Williams, "Squares: Supporting interactive performance analysis for multiclass classifiers," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 1, pp. 61–70, Jan. 2017.

- [33] B. Rieck and H. Leitte, "Enhancing comparative model analysis using persistent homology," in *Proc. Vis. Predictive Analytics Workshop*, 2014, http://predictive-workshop.github.io/papers/vpa2014_11.pdf, Accessed 09 Nov. 2018.
- [34] F. Roli, "Multiple classifier systems," *Encyclopedia Biometrics*, pp. 1142–1147, 2015, https://link.springer.com/referenceworkentry/10.1007%2F978-1-4899-7488-4_148
- [35] D. Sacha, M. Sedlmair, L. Zhang, J. A. Lee, J. Peltonen, D. Weiskopf, S. C. North, and D. A. Keim, "What you see is what you can change: Human-centered machine learning by interactive visualization," *Neurocomputing*, vol. 268, pp. 164–175, 2017.
- [36] C. Sammut and G. I. Webb, *Encyclopedia of Machine Learning*. Berlin, Germany: Springer, 2011.
- [37] B. Schneider, D. Jäckle, F. Stoffel, A. Diehl, J. Fuchs, and D. Keim, "Visual integration of data and model space in ensemble learning," arXiv preprint arXiv:1710.07322, 2017.
- [38] D. Shepard, "A two-dimensional interpolation function for irregularly-spaced data," in *Proc. 23rd ACM National Conf.*, 1968, pp. 517–524.
- [39] C. Silva and B. Ribeiro, "Visualization of individual ensemble classifier contributions," in *Proc. Int. Conf. Inf. Process. Manag. Uncertainty Knowl.-Based Syst.*, 2016, pp. 633–642.
- [40] J. Talbot, B. Lee, A. Kapoor, and D. S. Tan, "Ensemblematrix: Interactive visualization to support machine learning with multiple classifiers," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 2009, pp. 1283–1292.
- [41] D. H. Wolpert, "Stacked generalization," *Neural Netw.*, vol. 5, pp. 241–259, 1992.
- [42] Y. Lu, R. Garcia, B. Hansen, M. Gleicher, and R. Maciejewski, "The state-of-the-art in predictive visual analytics," in *Comput. Graph. Forum*, vol. 36, pp. 539–562, 2017.
- [43] T. Zhang and Q. Dai, "Hybrid ensemble selection algorithm incorporating grasp with path relinking," *Appl. Intell.*, vol. 44, no. 3, pp. 704–724, 2016.
- [44] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. Boca Raton, FL, USA: CRC Press, 2012.



Bruno Schneider received the MSc degree from the School of Applied Math, FGV, Rio de Janeiro, Brazil, in 2014. He is working toward the PhD degree at the Data Analysis and Visualization Group, University of Konstanz. His main research interests include information visualization, interactive machine learning and classification problems, in particular.



Dominik Jäckle received the PhD degree in visual analytics from the Data Analysis and Visualization Group, University of Konstanz. He develops, researches, and implements methods to make sense of vast amounts of multivariate data by combining interaction-rich interfaces with data mining and machine learning approaches.



Florian Stoffel received the MSc degree in information engineering from the University of Konstanz, in 2013. Since 2013, he works as a research associate with the Data Analysis and Visualization group, University of Konstanz. His research focuses on analytic and visual methods for analyzing text and intelligence data and methods to visualize their outcomes.



Alexandra Diehl received the PhD degree in computer science from the University of Buenos Aires, in 2016. She is a postdoctoral researcher with the Data Analysis and Visualization Group, University of Konstanz. Her main research interests include the area of social media analytics, with a special focus on VGI Science.



Johannes Fuchs received the PhD degree in computer science from the University of Konstanz. He is a research scientist and lecturer with the University of Konstanz. His main research interests include the area of visual analytics and information visualization with a special focus on data glyph design and evaluation.



Daniel Keim is a full professor and the head of the Data Analysis and Visualization Research Group, University of Konstanz. Keim received a habilitation in computer science from the University of Munich and has been program cochair of the IEEE InfoVis, the IEEE VAST, and the ACM SIGKDD Conference.