Visual Methods for Analyzing Probabilistic Classification Data

Bilal Alsallakh, Allan Hanbury, Helwig Hauser, Silvia Miksch, and Andreas Rauber

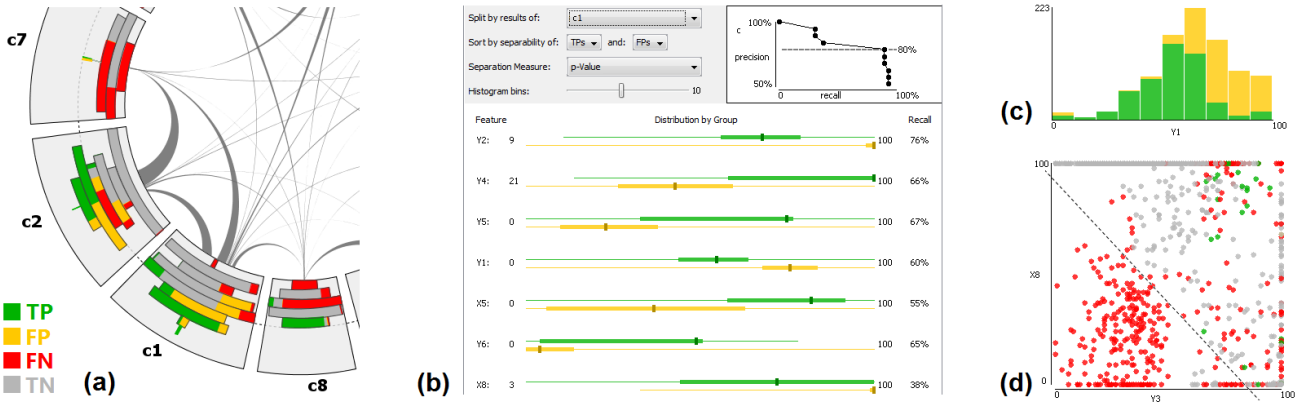


Fig. 1. Our visual analysis tools: (a) the *confusion wheel* shows sample-class probabilities as histograms colored by classification results, (b) the *feature analysis view* depicts feature distributions among selected samples, separated by their results, and ranked by a separation measure, (c, d) histograms and scatterplots reveal the separability of selected true and false classified samples by one or two features.

Abstract— Multi-class classifiers often compute scores for the classification samples describing probabilities to belong to different classes. In order to improve the performance of such classifiers, machine learning experts need to analyze classification results for a large number of labeled samples to find possible reasons for incorrect classification. Confusion matrices are widely used for this purpose. However, they provide no information about classification scores and features computed for the samples. We propose a set of integrated visual methods for analyzing the performance of probabilistic classifiers. Our methods provide insight into different aspects of the classification results for a large number of samples. One visualization emphasizes at which probabilities these samples were classified and how these probabilities correlate with classification error in terms of false positives and false negatives. Another view emphasizes the features of these samples and ranks them by their separation power between selected true and false classifications. We demonstrate the insight gained using our technique in a benchmarking classification dataset, and show how it enables improving classification performance by interactively defining and evaluating post-classification rules.

Index Terms—Probabilistic classification, confusion analysis, feature evaluation and selection, visual inspection

1 Introduction

The performance of classifiers in terms of correct classification is a major factor in determining their applicability for a given problem. Significant advances in machine learning have led to the development of a variety of classifiers and to an improved understanding of their properties. Designing a classification algorithm for a given problem is usually an iterative process that involves several decisions and choices. These include choosing an appropriate classifier, parameter tuning of this classifier, feature selection, and possibly introducing specific extensions to the classifier in order to handle special cases or to incorporate domain knowledge. This process aims to optimize the performance of the classifier according to some measures such as error rate, cost, or risk. For each of the above-mentioned stages in the design process, machine-learning experts need to understand the data involved in order to make choices that increase performance. Providing tools that assist these experts in analyzing the data in relation to the classification performance enables valuable guidance for the design process [20,41].

- *Bilal Alsallakh, Allan Hanbury, Silvia Miksch, and Andreas Rauber are with Vienna University of Technology. E-mail: {lastname}@ifs.tuwien.ac.at*
- *Helwig Hauser is with University of Bergen. E-mail: helwig.hauser@uib.no*

Manuscript received 31 Mar. 2014; accepted 1 Aug. 2014. Date of publication 11 Aug. 2014; date of current version 9 Nov. 2014.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.
Digital Object Identifier 10.1109/TVCG.2014.2346660

Visualization has played an important role in understanding and comparing classification algorithms and in improving their design (Sect. 2). In case of multi-class classifiers, the performance is usually reported by means of a confusion matrix that records for each class how many times its samples were confused for each other class (Fig. 2a). Compared with overall performance measures, these matrices provide more details about the results and help in introducing appropriate adjustments to the classifier. Besides predicting the class for a given input sample, many multi-class classification algorithms compute likelihood scores for a sample being of each of the classes. Analyzing how these scores correlate with the classification error and data features is important to understand the behavior of such classifiers. Confusion matrices discard this information as they incorporate final classifier decisions only. This paper presents visual methods for analyzing classification results of a multi-class probabilistic classifier for a large number of labeled samples. After motivating this problems and identifying related tasks (Sect. 3- 4), we describe how our methods allow analyzing classification results in context of class probabilities (scores) and data features. Our main contributions are:

- Involving class probabilities in the analysis of classification data by explicitly representing them as colored histograms.
- Intertwined automated and visual methods to analyze data features in relation with classification results and probabilities, and to rank them by their separation of true and false classification.
- An interactive exploration environment to analyze different aspects of probabilistic classification data.

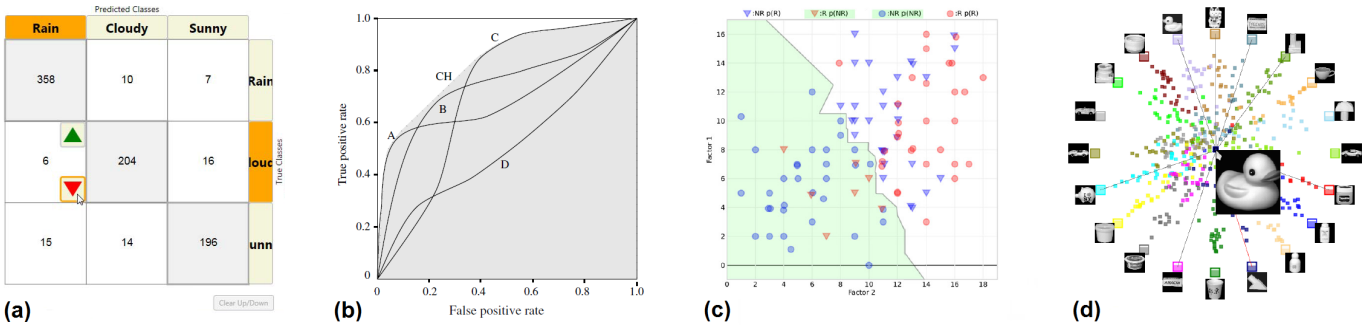


Fig. 2. Techniques for visualizing classification results: (a) an interactive confusion matrix [18], (b) ROC curves comparing five classifiers [13], (c) binary classification boundaries projected on two factors [25], (d) Class Radial Visualization [32].

Sect. 5 presents usage scenarios to demonstrate the applicability of our methods in analyzing and improving classification results. In Sect. 6 we discuss the advantages and shortcomings of our approach compared to previous work, and report expert feedback as well as practitioners experience in analyzing their classification data.

2 Related Work

A variety of approaches and tools have been proposed to improve classification performance using visualization. They can be categorized into techniques that engage the user actively in building the classifier, and those that focus on retrospective analysis of the performance.

2.1 Building Classifiers Interactively

Ware et al. [41] argue that classifiers built by users can compete with automated techniques as the users can incorporate their domain knowledge in the classifier design. Several techniques have been proposed for interactively constructing specific classifiers such as ones based on linear discriminant analysis (LDA) [9] or decision trees [4, 38, 40]. Also, similar ideas were proposed for specific aspects of classifier design such as distance measures [3, 7] and feature selection [6, 23, 43].

Certain techniques offer visual support for active learning, an approach which enables machine learning algorithms to query the user for the desired class of an unlabeled sample [33]. This learning paradigm has been shown useful in several domains such as video analysis [16] and document retrieval [15] where the number of samples is very large, prohibiting a manual labeling beforehand [31].

Talbot et al. [35] presented an interactive system to support ensemble learning, an approach to combine multiple classifiers to build one that is superior to its components. Their *EnsembleMatrix* technique visualizes the confusion matrices of the individual classifiers and allows combining these classifiers interactively with immediate update to the combined confusion matrix. Kapoor et al. [18] developed *ManiMatrix*, a system to refine a classifier by means of simple interactions with the confusion matrix (Fig. 2a). Reducing the tolerance for confusion between two classes triggers a search for new classification boundaries and updates the matrix interactively if a solution is found.

2.2 A Posteriori Analysis

Several visualization techniques were developed to help machine-learning experts analyze classification results *a posteriori*. These techniques are not tightly integrated with the classifiers, but are designed for post-mortem analysis, which makes them potentially classifier-agnostic. We describe next three categories of these techniques according to the primary information they visualize.

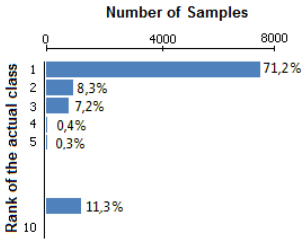
Classifier performance: Receiver operating characteristic (ROC) curves [13] and their variations [12] are well-established methods for tuning, assessing, and comparing the performance of binary classifiers. A ROC curve plots the true positive rate against the false positive rate of a binary classifier for a varying discrimination threshold (Fig. 2b). While ROC analysis can be extended to multi-class classifiers, it is still computationally exhaustive due to the large number of class combinations that need to be computed [20]. Also, visualizing high-dimensional ROC spaces is challenging, with existing techniques being able to show only partial information about the classes [14].

Data features: These techniques are dedicated to analyze the influence of data features on the classification results. Anand et al. [3] use a bubble chart to depict how the samples of a target class are distributed based on the values of one nominal and one numerical data features. Kienreich and Seifert [19] use feature-class matrices to show how features correlate with classes. A number of techniques visualize the decision boundaries of a binary classifier in a multi-dimensional feature space [8, 25]. Multiple scatter plots of the data features are used for this purpose (Fig. 2c). A recent follow-up technique augments the data points with information about their distances to the decision boundary [26]. This was shown useful for steering the classification model and identifying cost-changing data elements.

Class probabilities: Dedicated techniques have been proposed to visualize class probabilities in the case of probabilistic classification. Rheingans and desJardins used a heatmap to visualize the probability of a given class for each value combination of two features [29]. They account for a larger number of features by creating a 2D projection of the feature space. Iwata et al. [17] proposed a projection of class probabilities to visualize multiple classes in the same time. Projection-based techniques can preserve interesting structures in the high-dimensional space. Nevertheless, they might potentially result in complex visualizations that require good understanding of their properties and semantics to interpret correctly. Seifert and Lex [32] proposed a simplified technique for visualizing class probabilities. It places the classes on a circle and depicts the samples as points in this circle based on their class probabilities (Fig. 2d). These probabilities can be shown on demand for one sample as lines of varying thicknesses. The points are colored according to their predicted classes. In our work we also employ a radial layout for the classes, but use different visual abstractions and interactions as we explain next.

3 Motivation of our Work

When classifying samples using a probabilistic classifier, it is possible to infer for a wrongly-classified sample whether the actual class is the 2nd, 3rd or last guess (i.e. the rank of the actual class). The next chart shows a histogram of the ranks computed by a classifier for the actual classes of 10,992 samples. About 8.3% improvement on the classification rate is possible if the classifier would succeed on the 2nd guesses, e.g. by simple adjustments to the classifier or by using additional classification rules. On the other hand, 11.3% of the samples fail with low improvement chance with the current classifier. These samples are hard to separate from non-class samples.



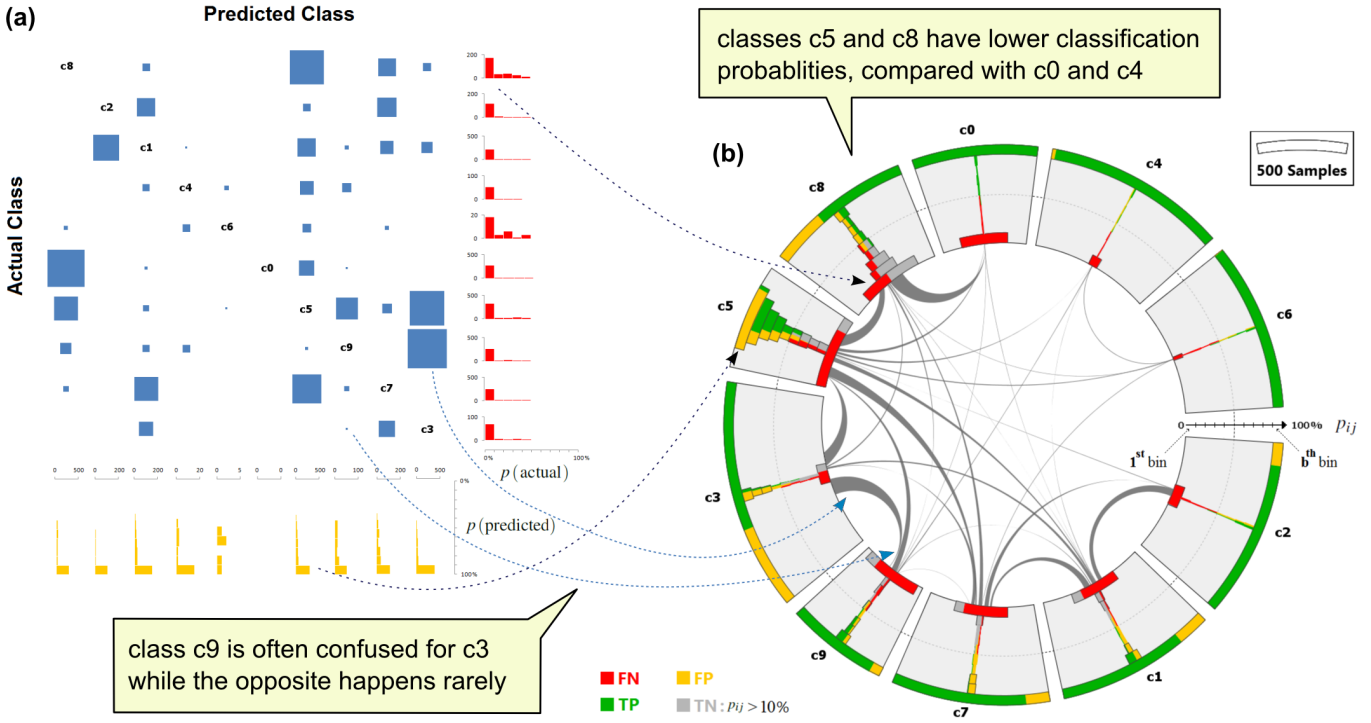


Fig. 3. Visualizing classification results of 10,992 handwritten digits [5] (a) using a confusion matrix augmented with histograms of sample probabilities in the respective rows and columns, (b) using the confusion wheel: Sectors represent digits with chords showing classification confusion between them. Histograms represent the probabilities of the samples in each class according to the color legend.

The histogram of actual class ranks does not provide actionable insight beyond indicating the amount of improvement potential. More detailed visualizations are needed to guide the users on how they can improve the performance. Fig. 3a shows a confusion matrix of the classification results described above. The size of a cell encodes the samples of its row class that are confused for its column class. The matrix is augmented with histograms of sample probabilities in each row and column. The row histograms represent false negatives (FNs), while the column histograms represent false positives (FPs) in the respective class. As we show in the next sections, this information is vital to understand the behavior of probabilistic classifiers. However, the matrix representation does not assign visual primacy to the histograms, which limits their usefulness. Moreover, it does not include information about correctly classified samples (true positive TPs and true negatives TNs) and how their probabilities are distributed, compared to misclassified samples. Finally, the information related to one class is scattered in multiple cells and two histograms in the respective row and column. We address these issues by employing alternative visual designs dedicated for analyzing probabilistic classification data.

4 Our Visual Analysis Tools

We propose a set of visualization tools that are integrated to analyze probabilistic classification data. The data encompasses:

- A set S of n labeled samples that are classified into m classes $C = \{c_{1 \le j \le m}\}$.
- The actual label for each sample $l^a(s) \in C : s \in S$.
- The predicted label for each sample $l^p(s) \in C : s \in S$.
- The probability p_{ij} for each sample $s_i \in S$ to belong to class $c_j \in C$, as computed by the classifier.
- A set of l data features $f_{1 \le k \le l}(s_i)$ for each sample $s_i \in S$, used by the classifier to compute the class probabilities.

The above information is available when classifying unknown samples, except for the actual labels l^a. Therefore, l^a-independent observations in the data can be reproduced during actual classifications.

Our tools aim to support the following analysis tasks:

- **T1:** Analyze the overall probability distribution of the samples to belong to a class (regardless to their actual classes).
- **T2:** Compare the probability distribution of the samples according to their classification results (TPs, FPs, FNs, TNs) in a class.
- **T3:** Find out FNs / FPs that have high / low probability. These samples are easier to improve than other FNs and FPs.
- **T4:** Select samples confused between two classes, and analyze their probability distributions in these classes.
- **T5:** Select samples by their class probabilities, classification results, or data features for further analysis.
- **T6:** Find out if FPs / FNs at a certain probability range can be separated from TPs / TNs in that range by the data features.

All of these tasks involve the class probabilities, and some of them involve the data features as well. Matrix representations fall short of supporting these tasks, as they assign visual primacy to class confusions. Therefore, we propose two main visualizations that assign visual primacy to the probabilities (Sect. 4.1) or to the data features (Sect. 4.2). We show in the next sections how these views are suited for solving the above-listed tasks, and enable new insight in the classification results beyond the information available in the matrix representation.

We illustrate our tools based on a UCI benchmarking dataset [5] that contains 10,992 labeled samples representing pen-based handwritten digits. The samples have 16 data features that comprise the x and y coordinates of eight points sampled along the curve of each digit.

4.1 Confusion Wheel

To assign visual primacy to the sample-class probabilities, we create histograms to show how they are distributed in each class (task **T1**). We employ the visual layout of Contingency Wheel++ [1] which places these histograms in a ring chart whose sectors represent the classes $c_1..c_m$ (Fig. 3b). In contrast to the matrix, this layout emphasizes the classes as primary visual objects with all information related to a class grouped in one place. This includes class probabilities and confusions with other classes as we explain next.

4.1.1 Visualizing sample-class probabilities as histograms

For each class c_j, the samples S are divided into four sets according to their classification results: TP_j, FP_j, TN_j, and FN_j. A histogram of the class probabilities is created for each of these sets using a uniform number of bins b, equal to n by default. The samples X_{jk} aggregated in bin k in this histogram are a subset of the corresponding set X_j, where X_j is one of the four sets mentioned above:

$$X_{jk} = \{s_i \in X_j : (k-1)/b < p_{ij} \leq k/b\} \quad (1)$$

A closed interval $[0, 1/b]$ is used for the first bin. The confusion wheel visualizes these histograms along the radial dimension in the respective sector, with the first bin placed next to the inner ring and the last bin b next to the outer ring. The user can select which histograms to include in the visualization (task **T2**). These histograms are stacked and centered in each sector to show the probability distribution of the respective samples. Color reveals the breakdown of these samples according to their classification results (Fig. 3b). All histograms have the same scale and the sectors are scaled to fit them.

By default, the confusion wheel filters out the bottommost bars of TNs having $p_{ij} \leq 10\%$. Showing these samples is of marginal interest, as they usually do not compete with the winner class. Filtering out these samples increases the resolution of the other histograms that show more important information about TPs and misclassified samples. Likewise, it is also possible to filter out the topmost bars of TPs having $p_{ij} \geq 90\%$ to further increase the resolution.

A reference circle indicates the 50% probability level in each sector. All negatives are located below this line. It is also possible for positives to lie below this line: this happens, for example, when the highest three class probabilities for a sample are nearly equal.

The colored histograms give more information about the classifier performance than confusion matrices. For example, it is evident in Fig. 3b that classes c_4 and c_6 have the clearest discrimination, with the majority of positive samples ($\geq 98\%$) in these classes being predicted with high probabilities ($\geq 90\%$). The opposite holds for c_5, where only 48% of its positive samples predicted with ($\geq 90\%$) probabilities. Only 25% of these samples were classified correctly. The percentage information can be obtained interactively in a tooltip.A naïve Bayesian classifier is used to classify the data.

Fig. 4 shows three classes from the data depicted in Fig. 3b. It includes only misclassified samples (FPs and FNs) depicted at a higher resolution by filtering out TPs and TNs (task **T3**). The samples are colored according to their actual classes. For this purpose, a unique color is assigned to each class from an appropriate qualitative color scale. As a result, the FNs in each class are colored by the class color. The FPs are colored by their actual classes, showing which other classes were confused for this class, and at which probability. This reveals that samples confused for c_3 are mostly of classes c_5 and c_9. Also, there is mutual confusion between c_5 and c_8 in the probability range $]0.2, 0.8]$. A misclassified sample s is double coded in Fig. 4 since it counts as a FP in $l^p(s)$ and as a FN in $l^a(s)$. This is indicated by the chords that represent class confusions are we explain next.

4.1.2 Visualizing class confusions as chords

The confusion wheel depicts class confusions as chords between the sectors (task **T4**). A chord between two classes is depicted with a varying thickness: the thickness at sector j_1 is proportional to $M_{j_2 j_1}$, the number of elements of class c_{j_2} confused for c_{j_1}. Likewise, the thickness at sector j_2 is proportional to $M_{j_1 j_2}$. Hence, following the chords outgoing from a sector reveals for which other classes its false negatives are confused. Alternatively, the chord can be split into adjacent ones that show the confusion in each direction individually. The sectors are ordered so that thicker chords are made shorter, using an $O(m^2)$ greedy algorithm [2]. This reduces the visual ink and the clutter caused by chord crossings, resulting in a clearer visualization. Also, this reveals groups of classes that have more confusion among each other than with the other classes. For example, it is evident in Fig. 3b that the digits 1, 2 and 7 are often confused with each other, as their shapes are similar to some degree.

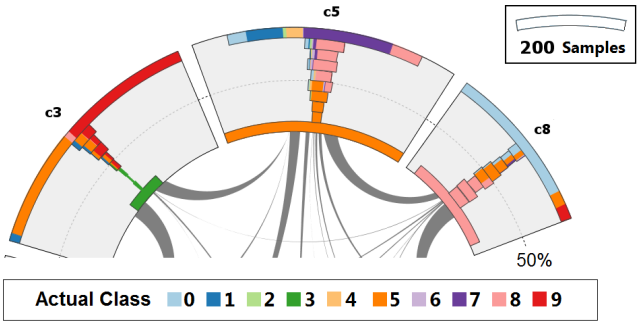


Fig. 4. Filtering and coloring the samples in the confusion wheel, representing the same data as in Fig. 3b. Only misclassified samples are shown (FPs and FNs), colored by their actual classes.

Compared with the chords, a matrix representation is more accurate at showing class confusions and their distribution in the matrix. Nevertheless, a chord is better at showing the mutual confusion between a pair of classes and the asymmetry of this confusion, compared with two visually-separate cells in a matrix.

4.1.3 Visualizing additional information about the samples

Instead of coloring the histograms by their classification results, a histogram bar can be alternatively colored by an attribute of the samples aggregated in it. For example, color can be used to compare the classification results against another classifier (Fig. 8). This shows for which samples the classification improved, worsened, or did not change in the depicted data (Sect. 5).

4.2 Feature Analysis View

Investigating the reason behind certain misclassifications and how to improve them depends heavily on analyzing how the data features are distributed among the affected samples. In particular, it is important to find out if certain features discriminate these samples from correctly classified samples. Therefore, we created a dedicated view that assigns visual primacy to the features by depicting how they are distributed in a selected subset of samples $\hat{S} \subseteq S$. As we did in the wheel view, we split the samples in $\hat{S}$ into the same four groups according to their classification result in a specific class $\bar{c} \in C$ selected by the user. To provide an overview first, we create up to four boxplots below each other for each data feature, showing how its values are distributed in each of the four groups $\hat{TP}_j$, $\hat{FP}_j$, $\hat{TN}_j$, $\hat{FN}_j$. If a group is empty, e.g. no FNs for $\bar{c}$ in $\hat{S}$, no boxplots are created for it. The view shows boxplots of multiple data features ordered in a list (Fig. 1b and 5g).

Typical feature analysis scenarios involve finding features that separate two main groups among selected samples: $\hat{TP}_j$ from $\hat{FP}_j$, or $\hat{TN}_j$ from $\hat{FN}_j$ (task **T6**). This has two implications on our design: First, we used a minimal version of boxplots, showing the whole value range, the median $Q2$, and the inter-quartile range $[Q1, Q3]$. We do not show outliers as they are irrelevant for the separation task. Second, and more importantly, we rank the features $f_{1 \leq k \leq l}$ by their separation power of two of selected groups X_1, X_2 from the above four groups. Several separation measures can be used for this purpose. One method is to use a significance statistic: for each feature f_k, we compute Welch's t-statistic [11] (which is used for Student's two-sample t-test with unequal sample sizes):

$$t_k = \frac{mean(f_k(X1)) - mean(f_k(X2))}{\sqrt{var^2(f_k(X1))/|X1| + var^2(f_k(X2))/|X2|}} \quad (2)$$

We rank the features by the corresponding p-values, computed according to Student's t-distribution. This places features with more significant mean differences between X_1 and X_2 in the top of the list. Such features are more likely to separate the samples in both groups, assuming the values in these groups are normally distributed as in Fig. 1c.

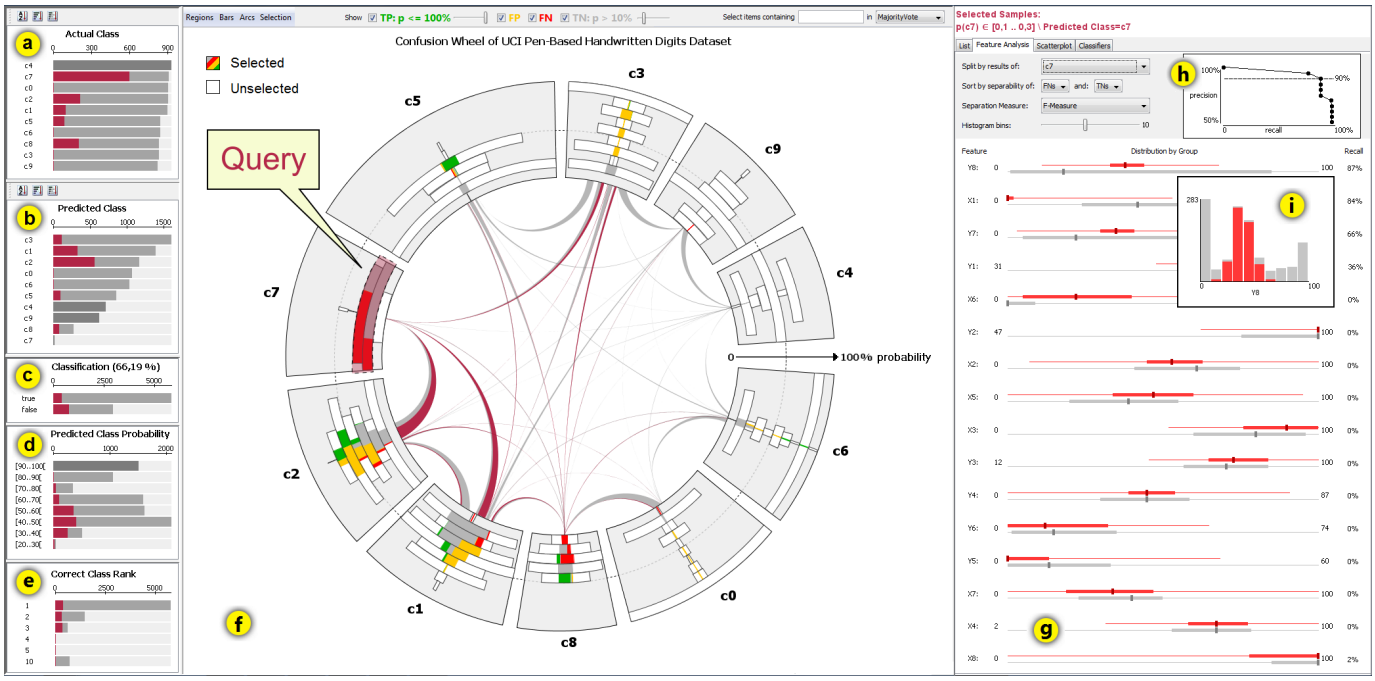


Fig. 5. The interactive exploration environment showing information about classification results, class probabilities, and feature distributions. The *summary charts* show breakdowns of the samples by (a) actual class, (b) predicted class, (c) classification correctness, (d) the probability of the predict class, (e) the probability rank of the actual class. The *wheel view* (f) shows the same data as in Fig. **??**b, classified using a *k*-NN classifier. Selected samples are highlighted in color. The *feature analysis view* (g) shows summary information data features of the selected samples, broken down according to their classification results, and the selection criteria in natural text. (h) A control panel to rank the features according to their separation power along with a recall-precision curve for possible separation. (i) A histogram of the top ranked feature.

Boxplots show only summary information of feature distribution. To gain more details about it, the user can click on a feature's area, which shows a stacked histogram of its values, depicting breakdown of the samples into the multiple groups described above. It helps in better estimating the separability of the groups by the selected feature.

In many cases, higher mean difference between two groups does not mean better separability. An example is shown in Fig. 5i, where two groups have relatively closer means, and yet better separability by the respective feature than by other features. This often happens when one of the groups represents combined phenomena like TNs that belong to different actual classes. To account for such cases, we provide alternative separation measures to rank the features. Both χ^2 and K-S statistics [22] are applicable generic measures to compare two empirical distributions without further assumptions. When defining additional classification rules (Sect. 5.4), it is important to identify *ranges* in feature distributions that have high separation. For this purpose, we use the following measure, based on the histograms h_{1k} and h_{2k} of a feature f_k in X_1 and X_2 respectively:

$$F_k = \sum_{b=1}^{b_h} h_{1k}(b) \cdot \frac{h_{1k}(b)}{h_{1k}(b)+h_{2k}(b)} \tag{3}$$

Similar to χ^2, this measure is computed from binned distributions with an adjustable number b_h of histogram bins. Instead of summing up deviations, it sums up the number of X_1 samples in each bin weighted by their retrieval precision, as with the F-measure [28].

To provide an overview of how much separation of X_1 from X_2 is possible using only one of the data features, we create a recall-precision graph in the top of the view (Fig. 5h). This graph indicates for each precision level to retrieve X_1 , the largest recall rate possible.

In some cases, no single feature provides good separation of the groups. Therefore, we offer a scatterplot view of selected samples $\hat{S}$ in the 2D space of two selected features, to check if these features offer better separation (Fig. 1d). Automated and visual techniques can be employed to recommend scatterplots with best separation [27, 30, 36]. In our implementation, the user selects the scatterplot dimensions from two lists of features, ranked by their univariate separability.

The visual tools described so far show different aspects of classification data. In the next sections we show how these tools are integrated together and describe use cases of our approach.

4.3 The Interactive Exploration Environment

Analysis scenarios of classification results typically involve examining different pieces of the information to formulate and test hypothesis about the results, and to introduce improvements. Therefore, we developed an exploration environment that shows these pieces of information at different levels of detail using multiple views that are arranged and coordinated accordingly in the user interface.

4.3.1 Summary views

These views show summary information about the classification results and performance. They assign visual primacy to the classification results, which are shown as secondary information in color in the other views. Each view highlights samples currently under selection. Three bar charts show breakdowns of the samples $s \in S$ by their actual class $l^a(s)$ (Fig. 5a), predicted class $l^p(s)$ (Fig. 5b), and classification correctness whether $l^p(s) = l^a(s)$ or not (Fig. 5c).

In addition, two histograms show breakdowns of the samples by the probability of the predicted class $p_{ij} : l^p(s_i) = c_j$ (Fig. 5d) and by the rank $r(s_i, l^a(s_i))$ of the actual class $l^a(s_i)$ (Fig. 5d), where:

$$r(s_i, l_j) = |\{1 \le j' \le m : p_{ij'} > p_{ij}\}| + 1 \tag{4}$$

4.3.2 Confusion wheel view

This is the central view in the user interface, showing aggregated information in more details than the summary views. Four checkboxes and two sliders at the top of this view enable quickly defining which samples to include. The sliders allow filtering TPs with high probability and FNs with low probabilities, to focus the analysis on more problematic samples to be depicted at a higher resolution (tasks **T3**).

Hovering the mouse over a visual element shows a tooltip with summary information about the samples that it represents (Fig. 6a). This encompasses, for example, recall and precision in a class, and the number of samples confused between two classes for a chord. More details about selected samples can be obtained in the detail views.

4.3.3 Detail and feature analysis views

The detail views show more information about the samples selected in other views. The top area in these views show a textual description of current selection. A tabular list shows the data features as well as the predicted and actual classes for the selected samples (Fig. 6b). The feature analysis view shows this information graphically, as explained in Sect. 4.2. Likewise, the *probability view* shows the class probabilities of the samples as a tabular list or as multiple histograms (Fig. 6c). The two tabular lists are synchronized: clicking on an sample in one view highlights it and ensures its visibility in both views. This enables a textual examination of the class probabilities of a certain samples. These probabilities are also depicted graphically as a star graph in the wheel view (Fig. 6a). Also, when possible, a graphical representation of this sample can be shown in a dedicated view (Fig. 6d).

4.3.4 Interactive Queries on the Samples [1]

By clicking on a bar in the summary views or in the wheel view, the respective subset $E \subseteq S$ of samples is selected (task **T5**). The views are immediately updated to highlight the fractions of bars and chords that represent elements in E. These fractions in the wheel view retain their colors. The rest of the elements become uncolored.

Multiple bars can be selected at once in a histogram in the wheel view. The selection in Fig. 5 is initially defined by brushing the range $[0.1, 0.3]$ over the radial dimension in c_7. This selects samples s_i with $p_{i,7} \in [0.1, 0.3]$. The selection is refined by excluding predicted samples in c_7, focusing only on TNs and FNs. These samples are highlighted in other sectors to show their classification results in the respective classes. The feature analysis view is updated to show the feature distributions among these samples.

The samples confused between two classes can be selected by clicking on the respective chord (task **T4**). This allows examining how these samples are distributed in the probability histograms of both sectors. The selection in Fig. 6a is defined by clicking on the chord between c_2 and c_7. The views are updated to show the class probabilities and feature values of the samples in E. These samples can be examined individually by clicking on an item in these views (Fig. 6c). Finally, samples can be further selected based on their features or other attributes by selecting a specific value range in the respective view.

The subsets that correspond to the above-mentioned selection possibilities can be combined interactively using set operations as in [2]. Specific keyboard modifiers allow specifying if the newly brushed elements should be added to, intersected with, or subtracted from the existing selection. This allows defining highly-expressive visual queries to select samples based on their classification results and probabilities in each class, and on their actual and predicted classes. For example, by clicking on c_3 in "predicted class" summary chart all positive samples in this class are selected (both TPs and FPs). This selection can be refined to TPs only by clicking on the respective bar in "actual class" while in set intersection mode. Also, certain FPs such as confusions with c_5 and c_9 can be filtered out by clicking on their bars in this chart while in set exclusion mode. We show in the next section how interactive selection of the samples supports several analysis scenarios of probabilistic classification data.

5 Usage Scenarios

We demonstrate how our tools can be applied to analyze and improve classification results of the UCI "pen-based" dataset introduced in Sect. 4. For this purpose, we train several classifiers using the raw features [2] of 100 randomly-selected samples using the RapidMiner data-

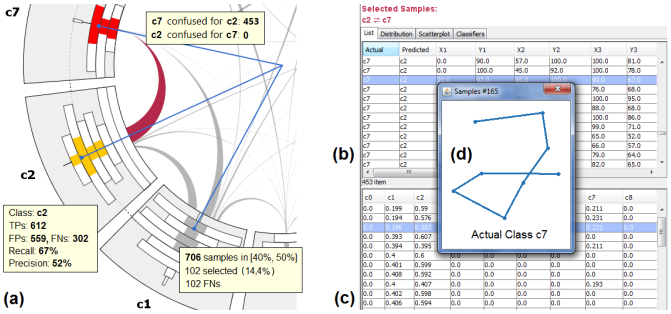


Fig. 6. Obtaining details about elements interactively: (a) selecting samples confused between c_2 and c_7. The list view show their (b) features and (c) class probabilities. One sample is selected for inspection by showing its graphical representation (d). Its class probabilities are highlighted both in the list view, and in the wheel view using arrows.

mining software [34] (formerly YALE [24]). We first show how interaction allows quick inspection of misclassified data. Then we show how the confusion wheel provides insight into classifier behavior, and enables comparing misclassified samples between two classifiers. Finally, we show how our tools help in defining additional classification rules to correct misclassified samples in a generalizable way. Besides demonstrating the standard features of our system, problem-specific features are introduced to support the last two use cases. Further examples with different data sets and classifiers are available at `http://www.cvast.tuwien.ac.at/ConfusionAnalysis/`

5.1 Inspecting Misclassified Samples

The rich possibilities to select subsets of samples using the interactive exploration environment allow quick inspection of certain samples, e.g., to analyze the reason behind certain confusions. In Fig. 6, elements confused between c_2 and c_7 are selected by clicking on the respective chord. Inspecting these samples illustrates that people write the same digits in different ways, which requires increasing the training sample to match against using k-NN classifiers.

In many cases, erroneous labels or noisy data features are the reason behind classification errors. Using our detail views we were able to identify such cases in the UCI dataset. One example is a c_5 digit confused for c_8 because it is written in east Arabic numerals which have different shapes than Arabic numerals. Another example was a noisy sample which does not resemble any digit. Identifying and isolating such samples is important to introduce effective design improvements and to accurately evaluate and compare different classifiers.

5.2 Analyzing Classifier Behavior

Visualizing the probability histogram for each class and coloring these histograms by classification results reveal several patterns that explain the behavior of the classifier. Fig. 7 shows this information for class c_5 using three different classifiers. With Neural Networks (NN), the classification accuracy increases proportionally with the probability of the predicted class (Fig. 7a). This does not always apply to a Naïve Bayesian (NB) classifier, where some classes showing the opposite trend (Fig. 7b). Also, though it varies between classes, the overall classification sharpness was higher for NB than NN, with 88.9% of all samples classified with $\geq 90\%$ probability (Fig. 3b), as opposed to 50.7% with NN. k-NN classifiers exhibit up to k peaks in the histograms at equidistant locations, when an appropriate number of bins b is used (Sect. 4.1). This is because k-NN classifiers perform weighted majority voting among the labels of the k nearest training samples to the samples being classified. If all k classifiers agree on the label c_j for s_i, p_{ij} is equal (or very close to) 1 and the sample belongs to the outermost peak in the histogram. If none of the classifiers agree, the label c_j of the closest training sample is predicted but with low probability, and the sample hence belongs to the innermost peak in the histogram of c_j. In Fig. 7c there are two peaks, as k equals 2. In Fig. 1a and Fig. 5f, there are up to five peaks per sector, as k equals 5.

[1] The supplementary video illustrates some of the interaction possibilities

[2] For reproducibility and illustration purposes, we did not consider computing any additional features that could improve the performance.

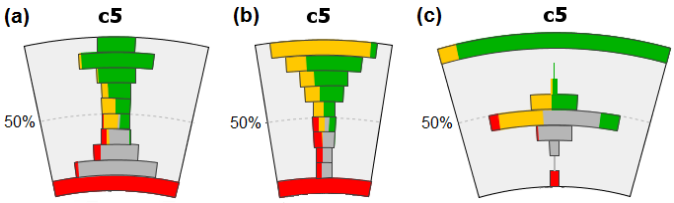


Fig. 7. Classification results in c_5 using (a) neural networks, (b) a naïve Bayesian classifier, and (c) a k-NN classifier with $k = 2$. TNs in the bottommost bars are filtered out. Individual histogram scales are used.

5.3 Comparison with Another Classifier

Classifier designers typically analyze the effect of using a different classification algorithm or changing certain parameters on the results. Besides a holistic measure of classification rate improvement, they often want to gain insight about the samples whose classification improved by this change, and the ones that worsened. A typical example is analyzing how the classes vary in their improvements, using two-sided bar charts that depict improved and worsened samples for each class in different directions. Another example is analyzing which class confusions increased or decreased by the changes, using a suited matrix representation. These representations discard available information on class probability which offers new ways to improve the performance. To address this limitation, we offer a mode to color the samples in confusion wheel by their improvement status as illustrated in Fig. 8. The data is classified using a k-NN algorithm with $k = 1$ and $k = 3$. The histograms show the class probabilities computed with $k = 3$. TNs are not shown, as they are irrelevant for comparison on class level. Dark blue indicates misclassified samples in the depicted data that would improve when $k = 1$. Dark red indicates correctly classified samples that would worsen when $k = 1$. It is noticeable that $k = 1$ performs better than $k = 3$ as there is more dark blue than dark red (overall 6% improvement). We investigated the reason for that by checking samples that improved or worsened. Fig. 8a illustrates an example of a sample whose nearest neighbor is the correct class, but the 2nd and 3rd nearest are not. In this example $k = 1$ succeeds while $k = 3$ fails. Fig. 8b shows the opposite case: 2 out of 3 nearest neighbors have the correct labels, making $k = 3$ succeeds and $k = 1$ fails. In both cases, the sample is close to the outer boundary in c_2 as two of the three nearest neighbors agree on its label.

Fig. 8c shows an interesting case which resembles Fig. 8a, with the only difference that the 2nd and 3rd neighbors are significantly far from the sample. This makes the classifier less confident about their votes, and hence predicting the answer at lower probability. Except for one sample, all 480 samples that fall in this probability range in this class would either improve or stay the same when $k = 1$. This suggests adding a rule to re-classify these samples, as we show next.

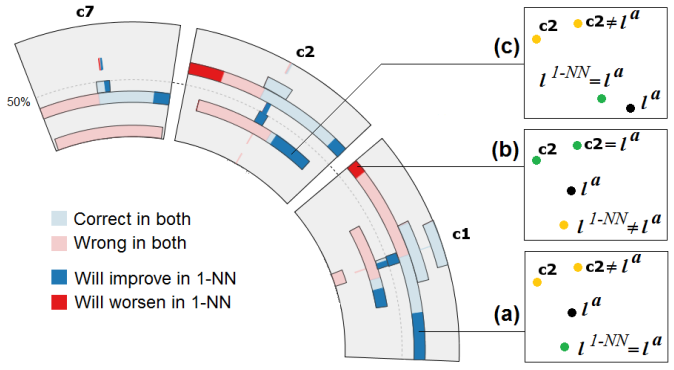


Fig. 8. Comparing the results of k-NN classifiers with $k = 1$ (encoded in color) against $k = 3$ (defining the histograms).

5.4 Defining Post-Classification Rules

In some cases, an existing classifier cannot be refined internally due to the lack of source code or expertise. Post-classification rules are one way to improve the performance in such situations by handling specific cases [39], incorporating domain knowledge [10], and rectifying systematic errors. Such rules are usually defined over the data features and are usually easy to understand and adapt especially if they are defined by domain experts. In case of probabilistic classification, a rule can specify Boolean conditions $q(s_i)$ on the class probabilities p_{ij} or ranks (Eq. 4) of the samples s_i, in addition to conditions on their features $f_k(s_i)$. Both types of information are available at runtime as they do not involve the actual labels $l^a(s)$.

We consider three rules for correcting classification errors:

- **Correcting false negatives:** This rule intends to correct FNs of class c_j by replacing their predicted classes with c_j:

$$\mathbf{R_{FN_j}} : q_1(s_i) \wedge \ldots \wedge q_k(s_i) \wedge l^p(s_i) \neq c_j \Rightarrow l^p(s_i) \leftarrow c_j \tag{5}$$

  Samples that satisfy conditions $q_1..q_k$ are post-classified as c_j.
- **Correcting false positives:** This rule intends to correct FPs of class c_j by replacing their predicted classes with the 2nd guesses:

$$\begin{aligned}\mathbf{R_{FP_j}} : q_1(s_i) \wedge \ldots \wedge q_k(s_i) \wedge l^p(s_i) = c_j \\ \Rightarrow l^p(s_i) \leftarrow c_{j'} : r(s_i, c_{j'}) = 2\end{aligned} \tag{6}$$

  It post-classifies a potential FP that satisfies its premise as the class $c_{j'}$ ranked 2nd for this sample (Eq. 4).
- **Using another classifier:** This rule intends to re-classify certain samples by using another classifier Cl_z:

$$\mathbf{R_{Cl_z}} : q_1(s_i) \wedge \ldots \wedge q_k(s_i) \Rightarrow l^p(s_i) \leftarrow Cl_z(s_i) \tag{7}$$

  It post-classifies a sample s_i that satisfies its premise as the class $Cl_z(s_i)$ predicted by Cl_z.

Our visual tools support in defining rules of the above types and in testing their actual improvement. For this purpose, the data set should be split into two parts: (1) training data that are loaded in the visualizations and used in defining the rules, and (2) validation data that are used to assess the actual improvement on unseen data. This is important to avoid dataset bias which occurs when defining rules that overfit the training data and fail to generalize to unseen data. Except for Fig. 3, the visualizations depicted in this paper use 80% of the UCI data introduced in Sect. 4. In the following we illustrate how potential improvements can be visually identified, and how the respective rules can be defined and validated on the remaining 20% of the data.

To improve misclassified samples in a class c_j, the respective rule should define conditions on the samples that include as much of these samples as possible and in the same time exclude correctly-classified samples. This is important as applying the rule on the latter samples would worsen their classification. The wheel view gives an overview on how misclassified samples c_j are distributed according to their probabilities. This makes it easy to spot probability ranges in c_j having a significant number of these samples that are likely to improve by one of the rules. Typically, these samples interfere with correctly classified samples. In particular, the interference of TPs (green) with FPs (yellow) requires separation using further conditions on the data features, before applying R_{FP_j}. This interference is usually larger in the outermost bin(s) that are dominated by TPs, which suggests excluding these bins when defining this rule. Similarly, the interference of FNs (red) with TNs (grey) requires separation in order to apply R_{FN_j}. Also the innermost bin(s) need to be excluded when defining this rule, as their probability range is highly dominated by TNs.

To separate the interference in a potentially improvable probability range Q_{c_j} in c_j, the user selects the samples in this range using brushing. The feature analysis view lists possible features that offer good separation, as explained in Sect. 4.2. After inspecting the feature histograms, the user can select a feature f_k to define an improvement

rule by double clicking on its histogram. This opens a dialog box which shows the histogram in higher resolution and allows selecting a specific value range Q_{f_k} for f_k (Fig. 9). The user selects a range that contains the majority of samples that need improvements (FPs or FNs) and excludes as much of the other samples as possible. The dialog also allows specifying which rule to apply to samples that fall both in Q_{c_j} and Q_{f_k}. The selected rule is externalized in text and applied to such samples both in the training dataset loaded in the visualization, and in the test dataset. The results in both cases are reported as the absolute number of samples that improved and worsened, and the total improvement on the classification rate. Changing the feature range Q_{f_k} automatically updates the results, to assist the user in choosing a robust range that performs well both in training and test datasets.

As example, in Fig. 1a, the analyst notices a large number of FNs in c_7. She selects the respective probability range in c_7 (Fig. 5f) and finds that feature y_8 offers good separation of these FNs from the TNs in the value range $[20\%, 60\%]$ (Fig. 5i). Therefore, she creates the following rule:

$$(0.1 \leq p_{i7} \leq 0.3) \wedge (20 \leq y_8(i) \leq 60) \Rightarrow l^p(s_i) \leftarrow c_7 \quad (8)$$

This rule improves 591 and worsens 13 samples in the loaded data, yielding a significant total improvement rate of 6.57%. Similar results apply to the testing data (141 improved, 3 worsened, 6.28% total rate) making the analyst accept this rule. She continues further to investigate the large number of FPs in c_1 by selecting the probability range $[30\%, 80\%]$ and restricting the selection to samples with $l^p(s) = c_1$. She checks the features for separation but notice interference between FPs and TPs, even with the features with most separation power (Fig. 9a). She selects the small range with as few TPs as possible, and applies rule R_{FP_1} which achieves 1.11% overall with 136 improved and 36 worsened samples in the training dataset. She rejects this rule and searches for more robust rules to improve these samples.

The scatterplot view allows identifying if two features can in combination achieve a good separation of interfering sample groups. As example, Fig. 1d illustrates how two features separate about 76% of FNs in c_8 that lie in probability range $[10\%, 30\%]$ from the TNs, with only 13 TNs unseparated. Reclassifying these samples with R_{FN_j} yields 3.5% and 3.4% improvements on the training and test datasets. Dedicated algorithms are needed to rank the pairs of features by their separation power, and to recommend optimal region separations in a specific scatter plot. For the purpose of this use case, a manual search for such features is sufficient to illustrate the importance of visual inspection to assess their separation power.

Besides searching for separating features, it is possible to improve certain misclassifications using the results of another classifiers. As example, it is evident in Fig. a that the results for c_2 involve a large number of mixed misclassifications (FNs and FPs) especially in the probability range $[30\%, 70\%]$. We provide a separate view to check how other classifiers perform on these samples by selecting them (Fig. 9b). This reveals that neural-networks-based classifier (NN) yields 7.31% improvement rate if applied to these samples, which suggests creating the following post-classifying rule (Eq. 7):

$$0.3 \leq p_{i2} \leq 0.7 \Rightarrow l^p(s_i) \leftarrow l^p_{NN}(s_i) \quad (9)$$

Such combinations of results from multiple classifiers has been extensively researched in pattern-recognition and machine-learning literature [21, 37, 42]. Many of these techniques apply combination heuristic such as weighted majority voting in a holistic way to all samples. We illustrated that declaratively restricting such heuristics to certain samples yields better improvements, as this avoid impacting correct classifications among the remaining samples. Using visual inspection, we were able to outperform automated techniques for combining multiple classifiers such as majority voting (Fig. 9b).

Post-classification rules should be defined carefully to avoid overfitting the data. First, the testing dataset should be representative and of sufficient size to warrant generalization. Second, the conditions used in these rules should be based on probability and feature ranges that have a sufficient number of samples to avoid creating rules specific to the training dataset. In fact, the distributions depicted in the probability histograms tend to be invariant among random subsets of sufficient sizes, if extreme values are discarded and avoided. Finally, defining several rules increases the overlap between their premises, the conflict in their actions, and the risk of over-fitting in general. It is important to select a small number of rules that exhibit robust improvement results, high precision, and few overlaps with each other. In general, classification errors should ideally be solved by improving the classification algorithm when possible, with help of the insights gained by the interactive visualizations.

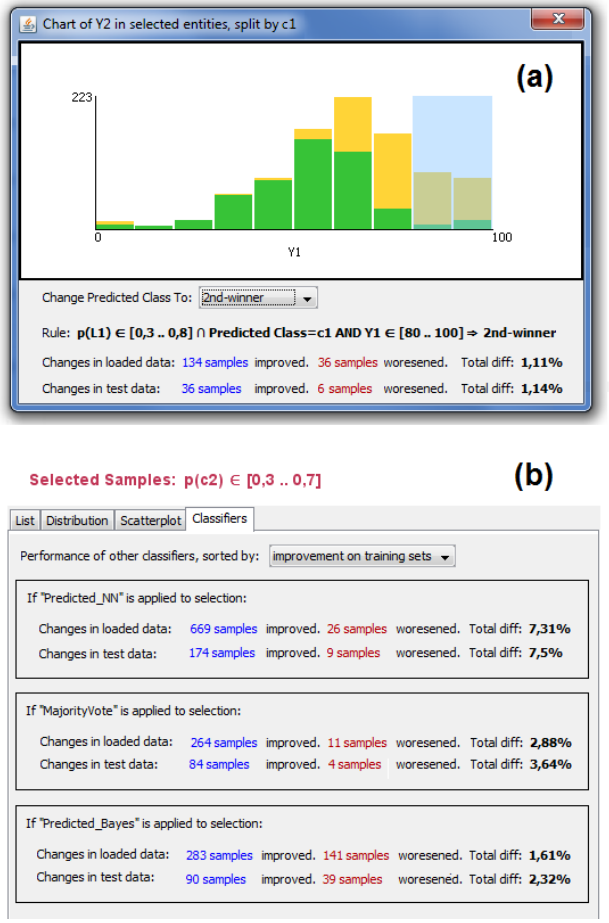


Fig. 9. Defining post-classification rules: (a) checking separability between FPs (yellow) and TPs (green) according to a feature, (b) checking the performance of other classifiers on selected samples.

6 Discussion

In this section we discuss the advantages and limitations of our tools and compare them with other techniques for visualizing probabilistic classifiers. We also report feedback and observations from classification experts and practitioners.

Scalability: The use of aggregated representations makes our tools highly scalable with the number of samples. For example, the histograms in the confusion wheel were able to handle datasets containing tens of thousands of samples. Furthermore, the filtering possibilities presented in Sect. 4.3 enable focusing on fine details contained in small subsets of these samples. Likewise, the boxplots and histograms in the feature analysis view scale well with the number of samples. Stacking the boxplots allow depicting summary information about 15-20 features at once. This is sufficient for our purposes, thanks to feature ranking which interactively places the most relevant features for the current analysis context at the top.

To ensure enough visibility of the information in each class, up to 20 classes can depicted at once as sectors in the wheel view. This

limit is feasible for a wide range of classification problems that do not require a larger number of classes. In case of larger number of classes, a subset of them can be selected manually or automatically to be shown at once, such as the subset with the highest confusions between its classes.

Handling imbalanced data: Sometimes, classification data exhibit skewed distributions of the samples to the classes. This causes classes having large number of positives to occupy the majority of display area, possibly obscuring fine details in smaller classes. To handle such cases is to possible to make the sectors of equal sizes, and stretch the histogram to fit in these sectors using individual scaling factors. Arcs representing the same amount of samples can be drawn outside the sectors using different scales to indicate these scaling factors. Although this hinders comparing the histograms in absolute values, the shapes of the distributions and the proportions of misclassified samples are still comparable across the classes. Such relative comparisons are usually more relevant in analyzing the results than comparing absolute values between classes of significantly different sizes. Similarly, the class confusions can be normalized by the total number of samples in the respective classes. These relative confusions can be indicated in color or by adjusting the chord thicknesses to show relative instead of absolute confusions.

Some classifiers compute relatively low probabilities for the winning class, such as ones based on fuzzy logic. This limits non-empty histogram bars to the inner bins only, which lowers the visual resolution. It is possible in such cases to redefine the bins to cover the effective probability range at higher resolution and make the histogram span the whole sector area.

Comparison with previous work: Our tools combine different pieces of information that have been addressed individually in previous work, as presented in Sect. 2. Compared with existing techniques that visualize class probabilities such as Class Radial Visualization (Fig. 2), the added-value in using the wheel metaphor lies in (1) aggregating the samples to analyze their probability distributions and to avoid clutter and ambiguity issues caused by individual points [32], (2) using color to show classification results of the samples next to each other and to reveal their separability, and (3) visualizing class confusions in a compact layout, thanks to the radial arrangement. This provides a rich overview of classification results that was not possible in previous techniques and enables new insight and tasks as we illustrate in the usage scenarios. Confusion matrices are better suited than the chords to gain more precise information about class confusions. Nevertheless, the chords reveal groups of classes having higher mutual confusions, emphasize the asymmetries in these confusions, and enable relating them to respective class probabilities.

The area-based nature of the histograms offers several possibilities for selecting and highlighting certain subsets of samples in confusion wheel. This is essential in a coordinated-multiple-view environment in order to formulate queries on the samples and to gain detailed insight into them in the feature analysis view. This environment enables a novel integration between probability-based and feature-based representations of classification data.

The visual analysis methods we propose are based on familiar visual representations such as boxplots and stacked histograms, as well as on the wheel metaphor [1]. This metaphor is originally designed to visualize associations between entities and categories, and similarities between categories. By treating samples as entities and classes as categories, we were able to adapt this metaphor for classification data: Instead of associations and similarities, we compute probabilities and two-way confusions. Also, we introduced several changes to the visual encoding and different interactions to address the characteristics of classification problems.

Expert feedback: We analyzed classification data comprising about $40,000$ labeled samples provided by our industry partners. The data were classified in 10 classes using a fuzzy rule-based system designed by domain experts. We refined our design iteratively based on feedback from these experts, and on what information they wanted to analyze. After we explained the visualizations we created for their data, they were able to identify issues with their classification rules. For example, one class had a large number of FNs that were highly concentrated in the middle of the respective sector. Analyzing the reason for that revealed that the respective classification rule was assigned relatively low weight by the domain experts. Also, examining the features for selected FPs in a class revealed a rule that uses an inappropriate feature that was mistaken for the correct one: The values of this feature among these FPs should not appear among actual samples of the class, according to domain experts. These experts were able to refine their system accordingly. We did not have the possibility to quantify the improvement.

The informal feedback from our industry partners and from five other machine-learning experts confirms that our tools offers both a good overview of classification results and detailed information on demand. However, our visual metaphors need to be learned with enough examples and explanations before they can be interpreted correctly: One domain expert, asked for clarification on what the histograms in the wheel view mean, about 30 minutes after we started presenting our findings. To avoid such misunderstanding, the metaphor should be introduced part by part with sufficient examples, before discussing insights. In fact, our system is feature laden, and needs extensive learning of how these features work together. One machine-learning expert requested showing separate arcs to encode class confusions in both directions. Also, two machine-learning experts did not encourage using post-classification rules in general, as they could encourage overfitting the data. They suggested using the insight gained in improving the classifier design instead.

7 Conclusion

The availability of class probabilities enables new possibilities to analyze the performance of probabilistic classifiers, beyond comparisons between predicted and actual classes. Common visual representations such as confusion matrices and ROC curves ignore class probabilities by assigning visual primacy to classification error in terms of false positives, false negatives, or class confusion. Assigning visual primacy to class probabilities or to the data features enables analyzing their influence on classification performance and performing further analysis tasks related to the data. We proposed a representation of probabilistic classification data by showing the probability distributions as stacked histograms in a radial layout and by coloring these histograms by the classification results of the samples. We showed how this representation lends itself to rich interactions to select samples based on their probabilities, and to perform further analysis of these samples based on their data features. We proposed intertwined automated and visual methods to analyze these features in a dedicated view and to rank them according to their separation power between true and false classifications among the selected samples. We presented several analysis scenarios that are possible using our visual tools. These include visual inspection and comparison of classification results, identifying performance problems, and interactive definition of post-classification rules to improve misclassified samples *a posteriori*. We demonstrated by that how exploratory analysis can reveal relevant patterns and correlations in classification data that are difficult to specify and identify automatically, and are usually compromised in holistic analysis methods. These insights are essential to introduce effective improvement to the classifier design that reduce the classification error in a generalizable way. Our future work aims to provide visual means to compare and combine the results of several classifiers, to support analyzing a large number of classes, and to explore hierarchical and multi-label classification results by means of similar interactive visualization methods.

Acknowledgement: We thank Peter Filzmoser, Colin Johnson, and Ammar Shaker for feedback and proposals, and Bilal Esmael and Arghad Aranout from TDE Data Engineering for cooperation and discussions. This work was supported by the Austrian Federal Ministry of Science, Research, and Economy via CVAST, a Laura Bassi Centre of Excellence (project no. 822746).

REFERENCES

[1] B. Alsallakh, W. Aigner, S. Miksch, and M. E. Gröller. Reinventing the contingency wheel: Scalable visual analytics of large categorical data. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12):2849–2858, 2012.

[2] B. Alsallakh, W. Aigner, S. Miksch, and H. Hauser. Radial sets: Interactive visual analysis of large overlapping sets. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2496–2505, 2013.

[3] A. Anand, L. Wilkinson, and D. N. Tuan. An L-infinity norm visual classifier. In *IEEE International Conference on Data Mining (ICDM)*, pages 687–692, 2009.

[4] M. Ankerst, C. Elsen, M. Ester, and H.-P. Kriegel. Visual classification: an interactive approach to decision tree construction. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 392–396. ACM, 1999.

[5] K. Bache and M. Lichman. UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences. `http://archive.ics.uci.edu/ml` (accessed: March 2014).

[6] S. Bremm, T. von Landesberger, J. Bernard, and T. Schreck. Assisted descriptor selection based on visual comparative data analysis. *Computer Graphics Forum*, 30(3):891–900, 2011.

[7] E. Brown, J. Liu, C. Brodley, and R. Chang. Dis-function: Learning distance functions interactively. In *IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 83–92, 2012.

[8] D. Caragea, D. Cook, H. Wickham, and V. Honavar. Visual methods for examining SVM classifiers. In *Visual Data Mining*, pages 136–153. Springer, 2008.

[9] J. Choo, H. Lee, J. Kihm, and H. Park. iVisClassifier: An interactive visual analytics system for classification based on supervised dimension reduction. In *IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 27–34, 2010.

[10] A. E. Daniels. Incorporating domain knowledge and spatial relationships into land cover classifications: a rule-based approach. *International Journal of Remote Sensing*, 27(14):2949–2975, 2006.

[11] J. Devore. *Probability and Statistics for Engineering and the Sciences*. Cengage Learning, 2011.

[12] C. Drummond and R. Holte. Cost curves: An improved method for visualizing classifier performance. *Machine Learning*, 65(1):95–130, 2006.

[13] T. Fawcett. An introduction to ROC analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[14] M. R. Hassan, K. Ramamohanarao, C. Karmakar, M. M. Hossain, and J. Bailey. A novel scalable multi-class ROC for effective visualization and computation. In *Advances in Knowledge Discovery and Data Mining*, pages 107–120. Springer, 2010.

[15] F. Heimerl, S. Koch, H. Bosch, and T. Ertl. Visual classifier training for text document retrieval. *Visualization and Computer Graphics, IEEE Trans. on*, 18(12):2839–2848, 2012.

[16] B. Hoferlin, R. Netzel, M. Hoferlin, D. Weiskopf, and G. Heidemann. Inter-active learning of ad-hoc classifiers for video visual analytics. In *IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 23–32, 2012.

[17] T. Iwata, K. Saito, N. Ueda, S. Stromsten, T. L. Griffiths, and J. B. Tenenbaum. Parametric embedding for class visualization. *Neural Computation*, 19(9):2536–2556, 2007.

[18] A. Kapoor, B. Lee, D. Tan, and E. Horvitz. Interactive optimization for steering machine classification. In *Proceedings of the International Conference on Human Factors in Computing Systems (CHI)*, pages 1343–1352. ACM, 2010.

[19] W. Kienreich and C. Seifert. Visual exploration of feature-class matrices for classification problems. In *International Workshop on Visual Analytics (EuroVA)*, pages 37–41. The Eurographics Association, 2012.

[20] N. Lachiche and P. Flach. Improving accuracy and cost of two-class and multi-class probabilistic classifiers using ROC curves. In *International Conference on Machine Learning (ICML)*, volume 20, pages 416–423, 2003.

[21] L. Lam and C. Y. Suen. Optimal combinations of pattern classifiers. *Pattern Recognition Letters*, 16(9):945–954, 1995.

[22] F. J. Massey Jr. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.

[23] T. May, A. Bannach, J. Davey, T. Ruppert, and J. Kohlhammer. Guiding feature subset selection with an interactive visualization. In *IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 111–120, 2011.

[24] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. YALE: Rapid prototyping for complex data mining tasks. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 935–940. ACM, 2006.

[25] M. Migut and M. Worring. Visual exploration of classification models for risk assessment. In *IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 11–18. IEEE, 2010.

[26] M. Migut, M. Worring, and C. Veenman. Visualizing multi-dimensional decision boundaries in 2d. *Data Mining and Knowledge Discovery*, pages 1–23, 2013.

[27] T. Pham, R. Metoyer, K. Bezrukova, and C. Spell. Visualization of cluster structure and separation in multivariate mixed data: A case study of diversity faultlines in work teams. *Computers & Graphics*, 38:117–130, 2014.

[28] D. M. Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2(1):37–63, 2011.

[29] P. Rheingans and M. desJardins. Visualizing high-dimensional predictive model quality. In *Proceedings of IEEE Visualization*, pages 493–496, 2000.

[30] M. Sedlmair, A. Tatu, T. Munzner, and M. Tory. A taxonomy of visual cluster separation factors. In *Computer Graphics Forum*, volume 31, pages 1335–1344. Wiley Online Library, 2012.

[31] C. Seifert and M. Granitzer. User-based active learning. In *IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 418–425, 2010.

[32] C. Seifert and E. Lex. A novel visualization approach for data-mining-related classification. In *Information Visualisation (IV), 13th International Conference*, pages 490–495, 2009.

[33] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[34] F. Shafait, M. Reif, C. Kofler, and T. M. Breuel. Pattern recognition engineering. In *RapidMiner Community Meeting and Conference*, volume 9, 2010.

[35] J. Talbot, B. Lee, A. Kapoor, and D. S. Tan. EnsembleMatrix: Interactive visualization to support machine learning with multiple classifiers. In *Proceedings of the International Conference on Human Factors in Computing Systems (CHI)*, pages 1283–1292. ACM, 2009.

[36] A. Tatu, G. Albuquerque, M. Eisemann, J. Schneidewind, H. Theisel, M. Magnor, and D. Keim. Combining automated analysis and visualization techniques for effective exploration of high-dimensional data. In *IEEE Symposium on Visual Analytics Science and Technology (VAST).*, pages 59–66. IEEE, 2009.

[37] D. M. Tax, M. Van Breukelen, R. P. Duin, and J. Kittler. Combining multiple classifiers by averaging or by multiplying? *Pattern recognition*, 33(9):1475–1485, 2000.

[38] S. T. Teoh and K.-L. Ma. PaintingClass: interactive construction, visualization and exploration of decision trees. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 667–672, New York, NY, USA, 2003. ACM.

[39] T. Van de Voorde, W. De Genst, and F. Canters. Improving pixel-based vhr land-cover classifications of urban areas with post-classification techniques. *Photogrammetric Engineering and Remote Sensing*, 73(9):1017, 2007.

[40] S. van den Elzen and J. van Wijk. Baobabview: Interactive construction and analysis of decision trees. In *IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 151–160, 2011.

[41] M. Ware, E. Frank, G. Holmes, M. Hall, and I. Witten. Interactive machine learning: letting users build classifiers. *Intl. Journal of Human-Computer Studies*, 55(3):281–292, 2001.

[42] L. Xu, A. Krzyzak, and C. Y. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *Systems, Man and Cybernetics, IEEE Transactions on*, 22(3):418–435, 1992.

[43] J. Zhang and L. Gruenwald. Opening the black box of feature extraction: Incorporating visualization into high-dimensional data mining processes. In *IEEE International Conference on Data Mining (ICDM)*, pages 1188–1192, 2006.