Case-Based Reasoning for Assisting Domain Experts in Processing Fraud Alerts of Black-Box Machine Learning Models

Hilde J.P. Weerts
h.j.p.weerts@student.tue.nl
Eindhoven University of Technology
Eindhoven, The Netherlands

Werner van Ipenburg
werner.van.ipenburg@rabobank.nl
Rabobank Nederland
Zeist, The Netherlands

Mykola Pechenizkiy
m.pechnizkiy@tue.nl
Eindhoven University of Technology
Eindhoven, The Netherlands

ABSTRACT

In many contexts, it can be useful for domain experts to understand to what extent predictions made by a machine learning model can be trusted. In particular, estimates of trustworthiness can be useful for fraud analysts who process machine learning-generated alerts of fraudulent transactions. In this work, we present a case-based reasoning (CBR) approach that provides evidence on the trustworthiness of a prediction in the form of a visualization of similar previous instances. Different from previous works, we consider similarity of local post-hoc explanations of predictions and show empirically that our visualization can be useful for processing alerts. Furthermore, our approach is perceived useful and easy to use by fraud analysts at a major Dutch bank.

KEYWORDS

explainable artificial intelligence, case-based reasoning, fraud detection, SHAP explanation similarity

Reference Format:
Hilde J.P. Weerts, Werner van Ipenburg, and Mykola Pechenizkiy. 2019. Case-Based Reasoning for Assisting Domain Experts in Processing Fraud Alerts of Black-Box Machine Learning Models. In *Proceedings of KDD Workshop on Anomaly Detection in Finance (KDD-ADF '19).*

1 INTRODUCTION

Machine learning models are increasingly applied in real-world contexts, including detection of fraudulent transactions. Often, the best performing models are complex models such as deep neural networks and ensembles. Despite their excellent performance, these models are not infallible, and predictions may require post-processing by human domain experts. However, as the complexity of the model increases, it can become more difficult for human domain experts to assess the correctness of a prediction. In such cases, domain experts who post-process predictions can benefit from evidence on the *trustworthiness* of the model's prediction. Moreover, this type of evidence could be useful to identify when the model is no longer trustworthy due to concept drift [3, 23], which is particularly relevant in the context of fraud detection. A straightforward indicator of the trustworthiness of a prediction is the model's own reported confidence score. However, raw confidence scores are often poorly calibrated [8], which means they can be misleading for human domain experts. Furthermore, since the models are not perfect, and in case of heavily imbalanced problems including fraud detection far from being perfect, even calibrated confidence scores can be inaccurate and hence misleading for processing of fraud alerts. Recent explanation methods, including e.g. SHAP [10], LIME [15], Anchor [16] can potentially help domain experts in determining to what extend the model's predictions can be trusted. For example, domain experts can look at local feature importance of the alert that is being processed or at a local surrogate model that mimics the behavior of the global model in the neighborhood of the alert. However, there is lack of empirical evidence that would illustrate the utility of such approaches for alert processing tasks. Our recent user study on the utility of SHAP for processing alerts suggests that SHAP explanations alone do not contribute to better decision-making by domain experts [21].

Approach. In the present paper, we introduce a case-based reasoning (CBR) approach to provide domain experts with evidence on the trustworthiness of a prediction. The proposed approach consists of two steps: (1) retrieve the k most similar instances to the query instance and (2) visualize the similarity as well as the true class of the retrieved neighbors. If the true class of similar instances corresponds to the prediction of the model, this provides evidence on the trustworthiness of the model's prediction, and the other way around. An important consideration of any nearest-neighbor type approach is the distance function. A straightforward notion of similarity in our scenario is similarity in feature values. However, instances with very similar feature values may be treated very differently by the model. Thus, it may be more useful for alert processing to consider similarity in *local feature contributions*. That is, we can consider whether the model's predictions of an instance can be explained in a similar way as the prediction corresponding to the alert. Different from previous works, we consider distance functions that take into account similarity in feature values, local explanations, and combinations thereof.

Empirical Evaluation. In simulated user experiments, we empirically show that our approach can be useful for alert processing. In particular, the usage of a distance function that considers similarity in local feature contributions often results in the best performance. Furthermore, a usability test with fraud analysts at a major Dutch bank indicates that our approach is perceived useful and easy to use.

Outline. The present paper is structured as follows. Section 2 covers related work. In Section 3, we introduce our CBR approach. In Section 4, we present the results of an empirical evaluation of our approach. We discuss concluding remarks in Section 5.

2 RELATED WORK

The basis of our approach is CBR: similar problems have similar solutions and previous solutions can be used to solve new problems [6]. CBR decision support systems became popular during the nineties and many different case-based explanations (CBE) have been suggested in that context [17].

Arguably the most straightforward CBE method is to retrieve the most similar cases. For example, Nugent and Cunningham [12] propose to retrieve the most similar instance to the current case, weighted by the local feature contributions of the query instance. The proposed distance function is intuitive, but its utility is not empirically tested. In the related field of k-Nearest Neighbor (k-NN) classification, the importance of the distance function has been long recognized. Different weight-setting algorithms have been proposed to improve the performance of k-NN algorithms. In particular, different algorithms can be distinguished based on whether weights are applied *globally* (i.e. feature importance across all instances) or *locally* (i.e. feature importance per instance or a subgroup of instances) [22]. In our experiments, we consider several distance functions, including unweighted, locally weighted, and globally weighted functions.

Similar to our goal, Jiang et al. [4] propose to capture trustworthiness into a *trust score*: the ratio between the distance from the query instance to the nearest class different from the predicted class and the distance to the predicted class. However, the utility of the trust score for human users is not evaluated. In particular, summarizing trustworthiness in a single score makes it impossible for users to determine whether the derivation of the score aligns with their domain knowledge. Moreover, the score can take any value, which can make it difficult to interpret by novice users.

3 CASE-BASED REASONING APPROACH

For a given instance, which we will refer to as the *query instance*, our approach consists of the following two steps (see Figure 1):

(1) *Case retrieval.* Retrieve the k instances from the case base that are most similar to the query instance. The appropriate distance function may be different for different problems. The case base consists of instances for which the true class is known.
(2) *Neighborhood visualization.* Visualize the k retrieved instances as well as the query instance as points in a scatter plot, such that:
    (a) the distance between any two instances corresponds to their similarity according to a distance function;
    (b) the colors of the neighbors correspond to their true classes.

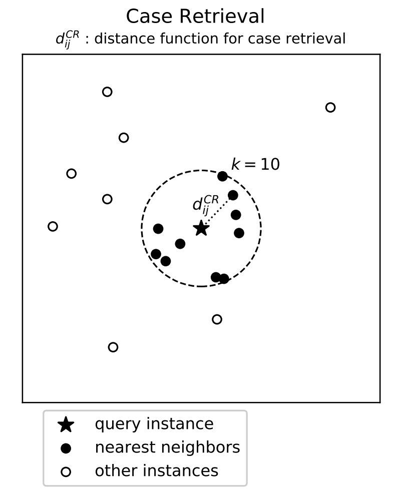
The number of retrieved cases, k, is a user-defined parameter; i.e. the user can choose how many instances they would like to retrieve. Note that a different distance function can be used for step (1) and (2). In Section 4.1, we empirically test which combination of distance functions are most useful in each step for several benchmark data sets as well as a real-life fraud-detection data set.

3.1 Case Retrieval

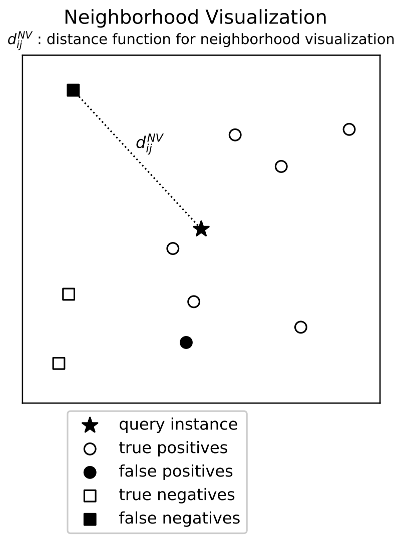
In the case retrieval stage, we retrieve instances from the case base that are similar to the instance we are trying to explain.

The *case base* consists of instances for which the ground truth is known at the time the machine learning model is trained. When the amount of historical data is relatively small, all instances can be added to the case base. Otherwise, sampling or prototyping approaches may be used to decrease the size of the case base.

Our approach assumes that the true class of instances in the case base is known. In some contexts, such as money laundering, the true class is typically not known for all instances. When instances whose true class has not been verified are added to the case base, this should be made explicit to the user in the neighborhood visualization, e.g. by means of a different color.



(a) ***Case Retrieval.*** **Retrieve the k instances most similar to the query instance.**



(b) ***Neighborhood Visualization.*** **Visualize how similar the retrieved k neighbors are as well as their true class.**

Figure 1: The two stages of the CBR approach for estimating the trustworthiness of a local prediction.

3.2 Neighborhood Visualization

Rather than just returning the retrieved neighbors to the user, we visualize the neighborhood in a two dimensional scatter plot (see Figure 1b). The distance between any two instances i and j in the visualization roughly corresponds to the dissimilarity of two instances, given by the used distance function. Similar to the approach taken by McArdle and Wilson [11], we compute the coordinates of the scatter plot using *multidimensional scaling* (MDS) [7].

In addition, colors or shapes can be used to visualize the model's performance for each of the retrieved instances. The user can use this information to determine whether the prediction of the query instance is trustworthy or not. For example, if many of the most similar neighbors are false positives, this decreases the trustworthiness of an alert.

The visualization can be further extended based on the application context. For example, we know that fraud schemes change over time. Hence, transactions that occurred a long time ago may be less relevant than newer transactions. In this scenario, a valuable extension could be to visualize the maturity of retrieved instances using e.g. an time-based color scale or a time filter. Another interesting extension would be to add *counterfactual instances* to the visualization. Counterfactual instances are perturbed versions of the query instance that received a different prediction from the model [20]. Adding such perturbed instances to the visualization may help the domain expert to identify and inspect the decision boundary of the classification model. We leave this to future work.

3.3 Distance Functions

The idea behind our approach is to retrieve and visualize instances that are similar to the query instance. However, it is unclear which notion of dissimilarity will be most useful for alert processing. Moreover, different combinations of distance functions in the case retrieval and visualization step may be more useful than others.

3.3.1 Feature Values. The most straightforward way to define similarity is to consider the feature values of transactions. In this case, instances that with similar feature values are considered similar. Depending on the feature type (e.g. categorical or continuous) different distance functions may be more appropriate than others. In this work, we assume that all feature values are properly normalized and that Euclidean distance is meaningful. However, we encourage readers to use a different distance function if appropriate.

3.3.2 Feature Contributions. A potential disadvantage of a plain feature value-based distance function is that the machine learning model is not taken into account at all. Instances that seem similar with regard to feature values, may have been treated very differently by the model. For example, consider a fraud detection decision tree with at its root node the decision rule `amount > $10,000`. Two transactions that are exactly the same with regard to all feature values except for *amount* are in different branches of the decision tree. Hence, the transactions may seem very similar in the data, but the decision-making process of the model could be completely different for each of the transactions. Judging the trustworthiness of a new prediction based on instances that were treated differently by the model does not seem intuitive. Instead, it might be more informative to take into account the model's arguments.

A state-of-the-art approach for explaining single predictions are Shapley Additive Explanations (SHAP) [10, 18]. SHAP is based on concepts from cooperative game theory and explains how each feature value of an instance contributed to the model's confidence score. In order to take into account the model's arguments for its predictions, we can consider similarity in SHAP explanations. In a SHAP value-based distance function, instances whose feature values contributed similarly to the model's confidence score are considered similar.

Interestingly, we find that distances in SHAP value space behave very well. First of all, SHAP explanations can be clustered well using relatively few clusters (see Figure 2b). Moreover, it is possible to identify subsets of transactions for which the model performs worse than for others (see Figure 2c). This indicates that SHAP value similarity can be meaningful for alert processing.

3.3.3 Globally Weighted Feature Values. A potential disadvantage of a distance function based solely on SHAP values, is that instances with a similar SHAP explanation can still have different feature values. We can combine SHAP explanations and feature values in a single distance function by weighting features values by the model's feature importance. Feature importance can be defined either globally (i.e. across all instances) or locally (i.e. per instance).

When considering a globally weighted distance function, instances with similar feature values on features that are considered *globally* important by the model (i.e. across all instances in the training data) are considered similar. Global SHAP feature importances can be computed as follows [9]:

$$|\bar{\Phi}| = \frac{1}{N} \sum_{i=1}^{N} |\Phi_i| \tag{1}$$

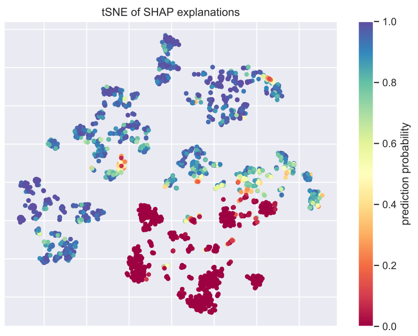
where N is the total number of instances in the data set and Φ_i the SHAP value vector of instance i.

3.3.4 Locally Weighted Feature Values. Local SHAP importances can be very different from global SHAP importances. For example, a particular feature may be relatively important on average, but not contribute at all to the prediction of a particular instance. Therefore, the utility of feature importance may depend on whether importance is defined globally or locally. When locally weighted feature value distance function is used, instances with similar feature values on features that are considered *locally* important by the model (i.e. for the query instance) are considered similar. Note that this distance function is similar to the one suggested by Nugent and Cunningham [12], except we use SHAP importances rather than local feature contributions similar in spirit to LIME.

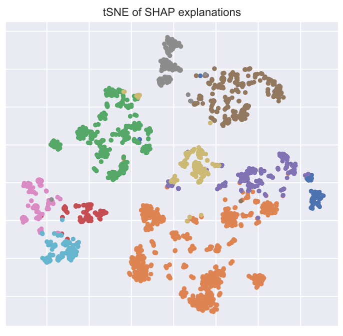
3.3.5 Formalization. As a basic distance function, we consider the weighted Euclidean distance. Given two input vectors $z_a = (z_{a1}, ..., z_{am})$ and $z_b = (z_{b1}, ..., z_{bm})$, and a weight vector $w = (w_1, ..., w_m)$, the weighted Euclidean distance is defined as follows:

$$d_{ab} = d(w, z_a, z_b) = \sqrt{\sum_{j=1}^{m} w(z_a - z_b)^2} \tag{2}$$

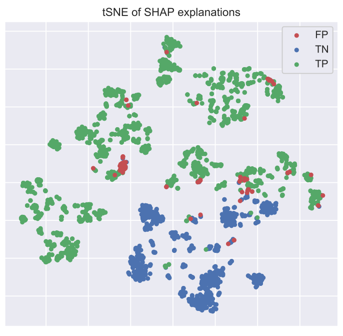
Note that Equation 2 is equivalent to the unweighted Euclidean distance when w is an all-ones vector ($\mathbf{1}$). We can describe the four considered distance functions with regard to the input vectors of Equation 2 (see Table 1).
In the next section we discuss the performance of our approach when d_F, d_S, d_G, and d_L are used. We will omit d and use the corresponding index letter in the figures for brevity.



(a) Model's Confidence



(b) k-means clustering $k = 10$



(c) Model's Performance

Figure 2: t-SNE visualization that groups transactions with similar SHAP explanations. The SHAP explanations explain predictions made by a random forest fraud detection model.

Table 1: In both the *case retrieval* and *neighborhood visualization* stage, we consider four distance functions that differ with regard to input values of Equation 2. x_a refers to the feature value vector of instance a, Φ_a to the SHAP value vector of instance a, and q denotes the query instance.

Notation	Description	Definition
d_F	feature values	$d(1, x_a, x_a)$
d_S	SHAP values	$d(1, \Phi_a, \Phi_b)$
d_G	feature values weighted by **global** SHAP importance	$d(\lvert\bar{\Phi}\rvert, x_a, x_b)$
d_L	feature values weighted by **local** SHAP importance	$d(\lvert\Phi_q\rvert, x_a, x_b)$

4 EVALUATION

The goal of our CBR approach is to provide evidence on the trustworthiness of a prediction. Similar to [4], we define *local trustworthiness* as the difference between the Bayes-optimal classifier's confidence and the model's prediction (e.g. fraud or no fraud) for that instance. That is, if the model agrees with the Bayes-optimal classifier, trustworthiness is high, and vice versa. Notably, even the Bayes-optimal classifier can be wrong in some regions due to noise. Hence, trustworthiness should be interpreted as an estimate of the reasonableness of the prediction, given the underlying data distribution. In practice, the Bayes-optimal classifier is not realizable and empirical measurements of trustworthiness are not possible. However, our goal is to provide decision support for domain experts that perform alert processing tasks. Hence, we can bypass the difficulty of measuring trustworthiness by measuring utility for domain experts instead.

4.1 Simulated User Experiment

We evaluate the expected utility of our visualization for alert processing in a *simulated user experiment.* In a simulated user experiment, assumptions are made about how users utilize an approach to perform particular tasks. Subsequently, the expected task performance is computed as the task performance that is achieved when applying the assumed strategy. Simulated user experiments have been used to evaluate the coverage, precision and trustworthiness of different explanations by Ribeiro et al. [15, 16]. We are not aware of simulated user experiments aimed at the evaluation of alert processing performance of domain experts. Consequently, we present a novel experiment setup.

In our simulated user experiments, we simulate a situation in which a machine learning model has been put in production. In this scenario, a model has been trained on historical data and new data is arriving. For each new instance, the model predicts whether it is a positive or a negative. Positives will trigger an alert and are inspected by human analysts, while negatives are not further inspected.

The goal of the experiments is to estimate how well a analyst would be able to process alerts when provided with the neighborhood visualization. To this end, we make a few assumptions about how users interpret the visualization. Based on these assumptions, we estimate how confident user would be about the instance belonging to the positive class. In order to determine the utility of the visualization compared to the model, we evaluate how well the estimated user confidence score as well as the model's confidence score correspond to the ground truth.

4.1.1 Method. To simulate the scenario where a model has been put into production, we split our data set in three different parts (see Figure 3). We can describe the simulated user experiment further using the following six steps:

(1) *Split the dataset.* We first split the dataset into three different sets: the *training data*, the *test data*, and *production data.* The production data represents data that arrives as the model is in production.
(2) *Train classifier.* We train a classifier on the training data.
(3) *Initialize Case Base.* As the training and test set contain instances for which we know the ground truth at the time the model goes into production, we add these instances to the *case base.*
(4) *Initialize Alert Set.* We determine which instances from the production data would result in a positive prediction from our machine learning model. These instances are put in the *alert set.*

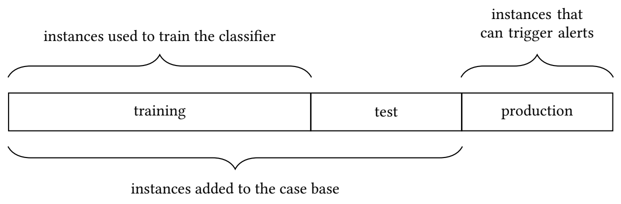

instances used to train the classifier
instances that can trigger alerts
training
test
production
instances added to the case base

Figure 3: In the simulated user experiment, the dataset is split into three sets. The *training* and *test* set correspond to data that is available at the time of model inference. The *production* set corresponds to new instances that arrive once the model is in production.

(5) *Estimate user's and model's confidence scores.* For each of the instances in the alert set, we estimate the user's confidence of the instance belonging to the positive class as a number between 0 and 1. Additionally, we determine the model's confidence for each instance in the alert set.
(6) *Evaluate confidence.* Given the ground truth of the instances in the alert set, we compare the mean average precision (MAP) that is achieved using the user's confidence to MAP achieved by the model's confidence.

In order to estimate the user's confidence, we make several assumptions on how our visualization is interpreted. Recall that we are interested in alert processing. We assume that a positive neighbor increases the user's confidence that the instance is a true positive and a negative neighbor decreases the user's confidence. Then, we can estimate the user's confidence, c_i based on the retrieved neighbors using the following equation:

$$c_i = \frac{1}{k}\sum_{j=1}^{k} \mathbf{1}\{y_j = 1\} \tag{3}$$

where y_j is the true class of instance j. However, some neighbors may be much more similar to the query instance than others. This is shown to the user in our neighborhood visualization, so the user will likely take this into account. Therefore, we weight each neighbor's class by the inverse distance between the neighbor and the query instance i:

$$c_i^w = \frac{\sum_{j=1}^{k} \frac{1}{d_{ij}} \mathbf{1}\{y_j = 1\}}{\sum_{j=1}^{k} \frac{1}{d_{ij}}} \tag{4}$$

Note that $\frac{1}{d_{ij}}$ is undefined if d_{ij} is equal to zero, i.e. if the neighbor is identical to the instance we are trying to explain. In our experiments, we deal with this by setting d_{ij} to a small number that is at least smaller than the most similar non-identical neighbor. In this way, a large weight to the identical neighbor, but the other neighbors are still taken into account.

4.1.2 Results. We evaluate our CBR approach on three benchmark classification data sets: *Adult* [5], *Phoneme* [1], and *Churn* [13]. All data sets were retrieved from OpenML [19]. Additionally, we evaluate our approach on a real-life *Fraud Detection* data set provided by a major Dutch bank. On each of the data set, we train a random forest classifier using the implementation in scikit-learn [14].

We evaluate the estimated user confidence scores for each possible combination of distance functions in Table 1. As a baseline, we also add a user confidence score that would be achieved when no distance function is considered in the neighborhood visualization (i.e. Equation 3). These results represent the case in which similarities are not provided to the user at all.

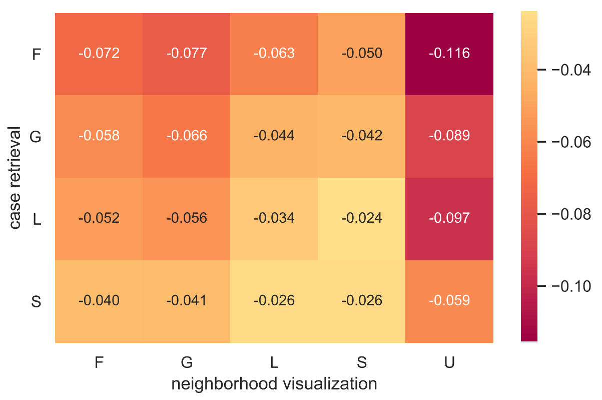
Recall that the number of neighbors k is a user-set parameter. Consequently, the approach is evaluated for different values of k, ranging from 1 to 500 neighbors. For each combination of distance functions, we compare the MAP of the model's confidence to the MAP of the estimated user confidence averaged over the different values for k. In Figure 4, we summarize the difference in performance as the average over all possible values of k.

Estimated User Confidence Mostly Performs Better Than Model's Confidence. For the *Churn*, *Phoneme* and *Fraud Detection* classification tasks, the estimated user confidence mostly results in higher average MAP than the model's confidence, but the achieved performance gain typically differs for different combinations of distance functions (Figure 4). Only for the *Adult* data set, the estimated user confidence results in worse average MAP scores than the model's confidence.

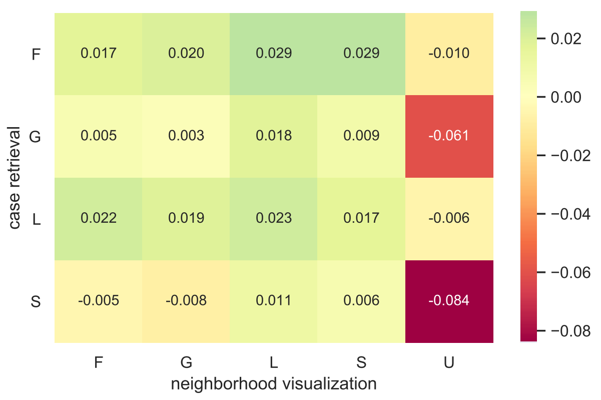
Number of Retrieved Neighbors (k) Impacts Performance. In some data sets, user-set parameter k has a high impact on the performance of the estimated user confidence. In particular, our approach outperforms the classifier in the *Adult* data set only for a very particular range of neighbors (see Figure 5a). For the *Phoneme* and *Fraud Detection* data sets, a minimum number of neighbors of approximately 20 is typically required to outperform the model's confidence score (see Figure 5b). This result suggests that returning only the most similar case to the user, as suggested by Nugent and Cunningham [12], may not provide enough evidence to be useful for alert processing. When applied to new problems, simulated user experiments could be performed to decide upon the appropriate range of k that can be selected by a real human user.

Unweighted User Confidence Performs Consistently Worse than User Confidence Weighted By Any Distance Function. For each of the data sets, estimating the user's confidence as the simple average of the true class of the retrieved neighbors consistently results in the worst performance. Recall that unweighted user confidence corresponds to a user who ignores similarity of the retrieved cases. This result shows the importance of communicating the similarity of the retrieved neighbors to the user, as is done in the *neighborhood visualization* step.

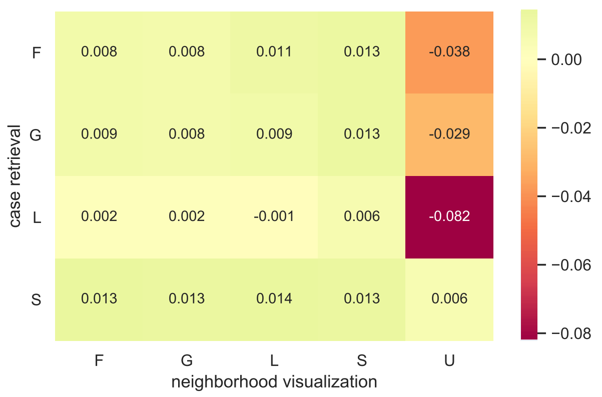
d_S Mostly Performs Best. For all data sets apart from the *Churn* data set, using d_S performs best for both case retrieval and neighborhood visualization (see Figure 4). In particular, performing case retrieval using d_S for the *Phoneme* and *Fraud Detection* data sets consistently results in top performance, regardless of the distance function that is used in neighborhood visualization. This indicates that the relevance of the retrieved neighbors is very high. In the *Churn* data set, d_F and d_L perform best for case retrieval and d_L for neighborhood visualization.



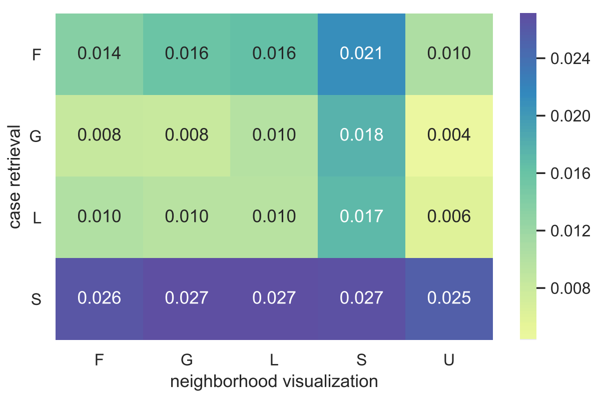
(a) ***Adult***



(b) ***Churn***



(c) ***Phoneme***



(d) ***Fraud Detection***

Figure 4: Improvement or decrease in average MAP of the estimated user confidence score compared to the MAP of the model's confidence score. MAP of the estimated user confidence is averaged over number of retrieved cases $k \in \{1, 2, ..., 500\}$. The difference is shown for all possible combinations of distance functions in the two steps of the approach. F, G, L, and S refer to the distance functions defined in Table 1. U refers to an unweighted estimated user confidence according to Equation 3 (i.e. if the user ignores the distances in the neighborhood visualization).

4.2 Usability Test

To determine the perceived utility of our approach for fraud analysts at the Rabobank, we conduct usability test.

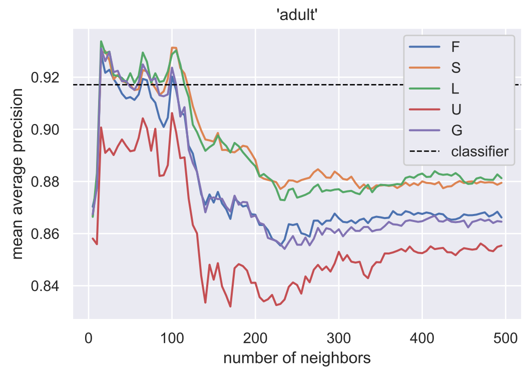
4.2.1 Method. The CBR approach is implemented in a Python-based dashboard, which displays the model's confidence, SHAP explanation, and neighborhood visualization of a selected alert (see Figure 6). The evaluation is performed in individual sessions with fraud analysts, using a think-out loud protocol. After the usability test, the dashboard is evaluated on *perceived usefulness* and *perceived ease of use* by means of a short survey introduced by Davis [2].

4.2.2 Results. Four fraud analysts participated in the evaluation. The average perceived usefulness was 5.64 on a 7-point Likert scale, with a standard deviation of 1.2. The average perceived utility was 5.96 out of 7, with a standard deviation of 0.9.

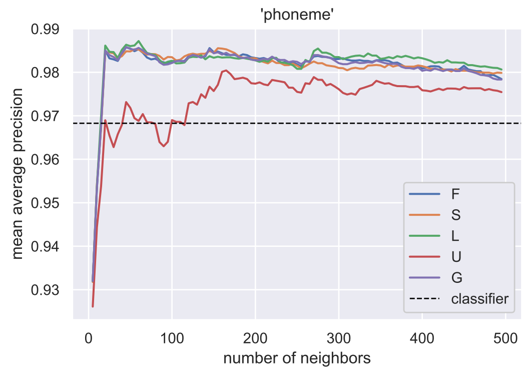
From the verbal protocols, it became clear that the neighborhood visualization materializes the fraud analysts' intuitions on the trustworthiness of fraud detection rules. As such, we expect the system to be particularly relevant for performing deeper analyses of cases for which a fraud analyst has not yet developed a strong intuition. As fraud detection models are constantly retrained, explanations for machine generated alerts are expected to differ over time, which makes our approach particularly relevant in that scenario.

5 CONCLUSIONS

Recent explanation methods have been proposed as a means to assess trust in a model's prediction. However, there is a lack of empirical evidence that illustrates the utility of explanations for alert processing tasks. In particular, understanding why the model made a certain prediction may not be enough to assess the correctness of the prediction. Hence, rather than explaining a prediction, our goal is to provide evidence on the reasonableness of the prediction given the underlying data. In this paper, we have introduced a novel CBR approach that can be used to assess the *trustworthiness* of a prediction.



(a) *Adult*



(b) *Phoneme*

Figure 5: The performance of different neighborhood visualization functions against the retrieved number of neighbors for the *Adult* and *Phoneme* data set when using d_S for case retrieval.

In our simulated user experiments, we have shown that the two-stage CBR approach can be useful for processing alerts. According to our intuitions, our results suggest that a distance function based on similarity in SHAP values is more useful than distances based on feature value similarity. Moreover, the results of a usability test with fraud analysts at a major Dutch bank indicate that our approach is perceived useful as well as easy to use.

5.1 Future Work

In the present paper, we have evaluated our approach on four different classification tasks with some varying results. Not all of these results are already well understood. In particular, future work could consider a more extensive analysis on why particular distance functions work well for some data sets and not as good for others.

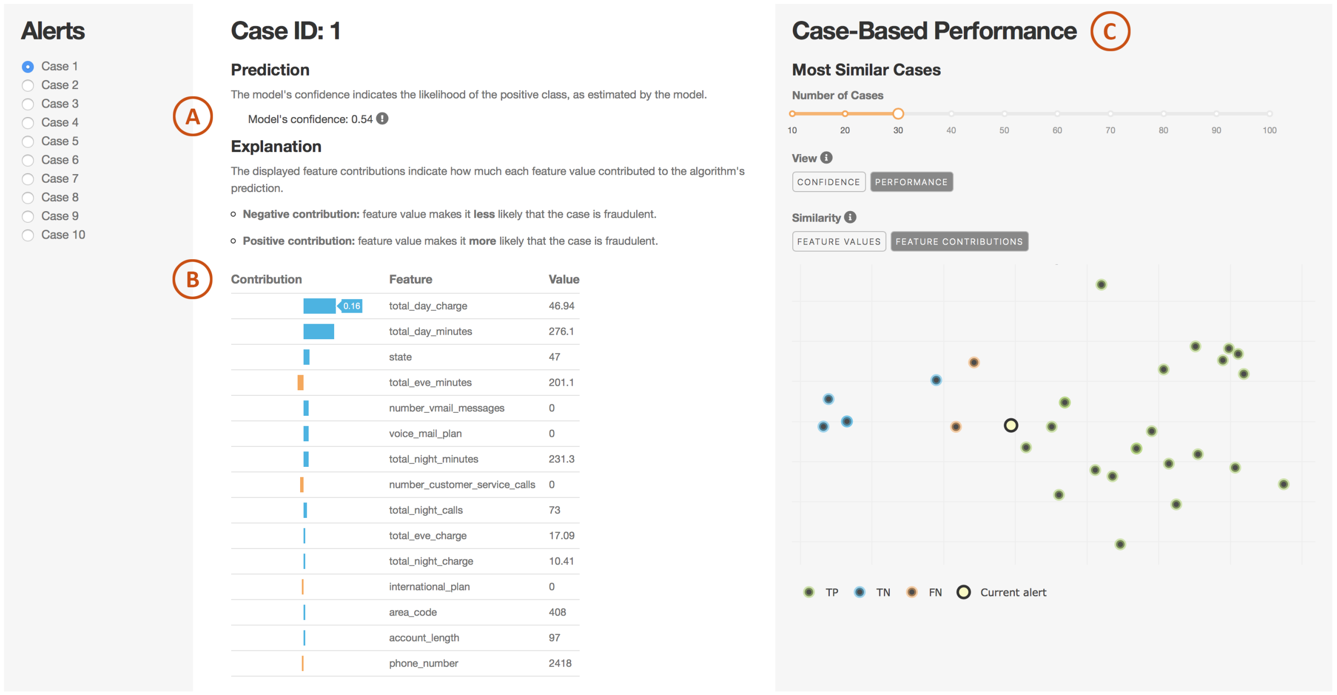
Additionally, future work could consider extensions of the neighborhood visualization. In particular, adding counterfactual instances is expected to provide more insights in the decision boundary of the model.

As SHAP values can be expensive to compute, future work could focus on optimizing the case base, by means of e.g. prototype selection or sampling approaches. An important aspect of these approaches is how they may be misleading for users. In particular, future work could study how over- and undersampling approaches affect the decision-making process of users in scenarios with highly imbalanced data.

One of the findings presented in this work is that SHAP explanations are remarkably clusterable. An interesting direction of future work that leverages this observation are *prototypical explanations*, which could be used to provide a global explanation of a black-box model in a model-agnostic fashion.

REFERENCES

[1] José Miguel Benedí Ruiz, Francisco Casacuberta Nolla, Enrique Vidal Ruiz, Inmaculada Benlloch, Antonio Castellanos López, María José Castro Bleda, Jon Ander Gómez Adrián, Alfons Juan Císcar, and Juan Antonio Puchol García. 1991. Proyecto ROARS: Robust Analytical Speech Recognition System.
[2] Fred D. Davis. 1989. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Quarterly* 13, 3 (1989), 319–340.
[3] João Gama, Indre Zliobaite, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *ACM Comput. Surv.* 46, 4 (2014), 44:1–44:37. https://doi.org/10.1145/2523813
[4] Heinrich Jiang, Been Kim, Melody Guan, and Maya Gupta. 2018. To Trust Or Not To Trust A Classifier. In *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.). Curran Associates, Inc., 5541–5552.
[5] Ron Kohavi. 1997. Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid. *KDD* (09 1997).
[6] Janet L. Kolodner. 1992. An introduction to case-based reasoning. *Artificial Intelligence Review* 6, 1 (1992), 3–34. https://doi.org/10.1007/bf00155578
[7] J. B. Kruskal. 1964. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika* 29, 1 (01 Mar 1964), 1–27. https://doi.org/10.1007/BF02289565
[8] Volodymyr Kuleshov and Percy S Liang. 2015. Calibrated Structured Prediction. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 3474–3482. http://papers.nips.cc/paper/5658-calibrated-structured-prediction.pdf
[9] Scott M. Lundberg, Gabriel G. Erion, and Su-In Lee. 2018. Consistent Individualized Feature Attribution for Tree Ensembles. arXiv:1802.03888
[10] Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems 30*, I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett (Eds.). Curran Associates, Inc., 4765–4774. http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf
[11] G.P. McArdle and D.C. Wilson. 2003. Visualising Case-Base Usage. In *Workshop Proceedings ICCBR*, L. McGinty (Ed.). Trondhuim, 105–114.
[12] Conor Nugent and Pádraig Cunningham. 2005. A Case-Based Explanation System for Black-Box Systems. *Artificial Intelligence Review* 24, 2 (oct 2005), 163–178. https://doi.org/10.1007/s10462-005-4609-5
[13] Randal S. Olson, William La Cava, Patryk Orzechowski, Ryan J. Urbanowicz, and Jason H. Moore. 2017. PMLB: a large benchmark suite for machine learning evaluation and comparison. *BioData Mining* 10, 1 (11 Dec 2017), 36. https://doi.org/10.1186/s13040-017-0154-4
[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
[15] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIG International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM Press, New York, New York, USA, 1135–1144. https://doi.org/10.1145/2939672.2939778
[16] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-Precision Model-Agnostic Explanations. In *AAAI*.
[17] Frode Sørmo, Jörg Cassens, and Agnar Aamodt. 2005. Explanation in Case-Based Reasoning–Perspectives and Goals. *Artificial Intelligence Review* 24, 2 (oct 2005), 109–143. https://doi.org/10.1007/s10462-005-4607-7
[18] Erik Štrumbelj and Igor Kononenko. 2014. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems* 41, 3 (2014), 647–665. https://doi.org/10.1007/s10115-013-0679-x



Alerts

- ● Case 1
- ○ Case 2
- ○ Case 3
- ○ Case 4
- ○ Case 5
- ○ Case 6
- ○ Case 7
- ○ Case 8
- ○ Case 9
- ○ Case 10

Case ID: 1

Prediction

A

The model's confidence indicates the likelihood of the positive class, as estimated by the model.

Model's confidence: 0.54

Explanation

The displayed feature contributions indicate how much each feature value contributed to the algorithm's prediction.

- **Negative contribution:** feature value makes it **less** likely that the case is fraudulent.
- **Positive contribution:** feature value makes it **more** likely that the case is fraudulent.

B

Contribution	Feature	Value
0.16	total_day_charge	46.94
	total_day_minutes	276.1
	state	47
	total_eve_minutes	201.1
	number_vmail_messages	0
	voice_mail_plan	0
	total_night_minutes	231.3
	number_customer_service_calls	0
	total_night_calls	73
	total_eve_charge	17.09
	total_night_charge	10.41
	international_plan	0
	area_code	408
	account_length	97
	phone_number	2418

Case-Based Performance

C

Most Similar Cases

Number of Cases: 30

Figure 6: CBR dashboard when applied to predictions of a random forest model trained on the *Churn* dataset. (A) The model's confidence for the selected alert, (B) A bar chart showing the SHAP values of the selected alert, (C) The CBR neighborhood visualization. The number of neighbors k can be chosen in the slider.

[19] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. 2013. OpenML: Networked Science in Machine Learning. *SIGKDD Explorations* 15, 2 (2013), 49–60. https://doi.org/10.1145/2641190.2641198

[20] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2018. Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR. *Harvard journal of law & technology* 31 (04 2018), 841–887.

[21] Hilde J.P. Weerts, Werner van Ipenburg, and Mykola Pechenizkiy. 2019. A Human-Grounded Evaluation of SHAP for Alert Processing. In *Proceedings of KDD Workshop on Explainable AI (KDD-XAI '19)*.

[22] Dietrich Wettschereck, David W. Aha, and Takao Mohri. 1997. A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. *Artificial Intelligence Review* 11, 1/5 (1997), 273–314. https://doi.org/10.1023/a:1006593614256

[23] Indre Zliobaite, Mykola Pechenizkiy, and Joao Gama. 2016. An overview of concept drift applications. In *Big Data Analysis: New Algorithms for a New Society*. Springer, 91–114.