# Explaining Probabilistic Fault Diagnosis and Classification using Case-based Reasoning

Tomas Olsson[1,2], Daniel Gillblad[2], Peter Funk[1], and Ning Xiong[1]

[1] School of Innovation, Design, and Engineering, Mälardalen University
Västerås, Sweden
[2] SICS Swedish ICT
Isafjordsgatan 22, Box 1263, SE-164 29 Kista, Sweden

**Abstract.** This paper describes a generic framework for explaining the prediction of a probabilistic classifier using preceding cases. Within the framework, we derive similarity metrics that relate the similarity between two cases to a probability model and propose a novel case-based approach to justifying a classification using the local accuracy of the most similar cases as a confidence measure. As basis for deriving similarity metrics, we define similarity in terms of *the principle of interchangeability* that two cases are considered similar or identical if two probability distributions, derived from excluding either one or the other case in the case base, are identical. Thereafter, we evaluate the proposed approach to explaining the probabilistic classification of faults. We show that with the proposed approach, it is possible to find cases for which the used classifier accuracy is very low and uncertain, even though the predicted class has high probability.

## 1 Introduction

Several papers from the last decades identify an intelligent system's ability to explain its predictions as a key factor for user acceptance [1–5]. Hence, a decision support system is less likely to be accepted if a user does not understand or trust its predictions or recommendations. For instance, in the medical domain, the physicians will not trust a system only because of good prediction performance but only if they understand the reasoning behind [6].

In a previous paper, we have proposed using case-based reasoning (CBR) as an intuitive approach for justifying (explaining) the predictions of a probabilistic model [7]. The idea is to support a non-expert user in assessing the system reliability by querying a CBR system for justifying explanations in form of a list of relevant, preceding cases, together with some sort of summary. While our previous work addressed the problem of explaining regression, that is, numerical predictions, this paper extends this approach to probabilistic classification, and specifically, for explaining fault diagnosis.

Fault diagnosis is about detecting when a fault occurs, its location and thereafter identifying the fault type and severity [8, 9]. Both CBR and model-based machine learning approaches have been applied to fault diagnosis [10–12]. Yet, traditionally, CBR is not used when there is a sufficiently good model-based solution to a problem. Still, CBR is conceptually simpler and arguable more intuitive than many model-based approaches, and thus, a case-based explanation facility can make the classification of faults more understandable.

This work is inline with previous work in CBR that uses cases to explain model-based machine learning algorithms [13, 14]. The problem of explaining model-based algorithms with cases is twofold. First, in order to relate cases to the learned model, a similarity metric that measures the usefulness of a case relative to the model is needed. Since in many cases the model was not defined with this in mind, this is not a straightforward problem to solve. Second, a method for explaining the prediction based on the cases must be developed. Considering that the prediction is done with the learned model, it is also reasonable that the explanation is closely related to the model. This work makes therefore two contributions in order to solve this problem.

The first contribution of this paper is to use the generic, theoretically well-defined approach to defining similarity metrics presented in [7] to probabilistic classification. As basis for the definition of similarity, we have formulated *the principle of interchangeability* that two cases are similar or identical if they can replace each other with respect to the probability model and a statistical measure of similarity [7]. In the previous paper, we modeled cases using log-normal linear regression, while in this paper, we use logistic regression.

The second contribution is a novel approach for explaining classification predictions in form of the *local accuracy*. In [7], we used the local mean absolute error to explain regression. The local accuracy is the fraction of the most similar preceding cases that are correctly classified. We interpret the local accuracy as an estimation of how likely it is that the system's prediction is correct. The local accuracy together with a list of preceding cases is then used as a justification of the system performance.

The rest of the paper is organized as follows. Sect. 2 presents related work. In Sect. 3, we give some background to similarity metrics, statistical metrics and logistic regression. Sect. 4 presents the overall framework for explanation and four derived similarity metrics. Sect. 5 describes the application of the proposed approach to explaining classification of faults. In Sect. 6, we make concluding remarks and describe future work.

## 2 Case-based Explanation

This section presents related work that – similarly to the proposed approach – uses cases for explaining systems. This is a research field called case-based explanation (CBE) [15–17, 4]. CBE can, similarly to CBR, be divided into knowledge intensive and knowledge light CBE where the former makes use of explicit domain knowledge while the latter uses mainly knowledge already contained in the similarity metric and the case base [18]. The current work is an instance of knowledge light CBE with no explicit explanation model.

Furthermore, knowledge light CBE differs in how cases are explained. While our work uses CBR to make explanations of model-based machine learning algorithms, other work uses model-based methods to make explanations of CBR systems. The ProCon system described in [19, 20] uses a naive Bayes classifier trained on all cases to find which features of a case support or oppose a classification. The system presented in [21] by the same author generates rules from the nearest neighbors in order to explain the retrieved cases. Both of these systems investigate and present information to the user on what in the preceding cases support and oppose the classification. This is not considered in our approach.

A second type of research investigates which cases to present to a user as an explanation. The similarity metric that was used for classification might not be the best for explanation. In [22], the authors compare similarity metrics optimized for explanation with those optimized for classification, while in [23], the authors use the same similarity metric used for classification but explore different rules for selecting which case to use as an explanation. In [24], logistic regression is used to find cases, close to the classification border, that are assumed to better explain a classification. In comparison, the current work does not use CBR for classification, but for explanation, and we assume that the similarity metric that is best for explanation is also the best for estimating the local accuracy.

The third type of knowledge light CBE research addresses – similarly to our approach – the explanation of model-based machine learning methods using cases [13, 25, 14, 26, 27]. The first knowledge light CBE for model-based learning algorithms was presented in [13]. In this paper, the author sketches ideas on how to use the model of a neural network or a decision tree as a similarity metric. In case of neural networks the activation difference between two cases was proposed as a metric while the leaves in the decision tree naturally contain similar cases. The neural network activation metric resembles our approach in that model parts are compared, but in contrast to our approach, the metric is not theoretically well-defined. A generic CBE framework for black-box machine learning algorithms is presented in [14]. A neural network was locally approximated using a locally weighted linear model based on artificial cases generated from the neural network. Then, the coefficients of the linear model were used both as feature weights of a similarity metric and for identifying the important features of a prediction. In case of our approach, there is no need to approximate the machine learning model, since only probability distributions are compared. In addition, the similarity metrics in our work are theoretically well-founded [7], while there is no theoretical motivation in [14] to why the linear regression weights can be used in a similarity metric.

## 3 Preliminaries

In this section, we define the notion of a true metric that is important in order to index cases for fast retrieval and we discuss the relation between similarity and a distance. In addition, we present the J-divergence that is a statistical measure of similarity between probability distributions that we use in our definition of similarity between cases. Last, we describe multinomial logistic regression that we use as an example of a probabilistic classifier for classifying faults.

### 3.1 Similarity and True Metrics

In order to make fast retrieval of cases possible, similarity metrics should adhere to the axioms of a true metric. Given a true metric, the search space can be partitioned into smaller regions and organized so that there is no need to search through all regions. A true metric is a distance or a dissimilarity metric, while in CBR, we typically talk about the similarity between cases. However, similarity and distance are coupled concepts in that a distance can easily be transformed to a similarity or the other way around. So, we will not make a precise distinction between distance metrics and similarity metrics in this work.

In this paper, we use the term metric informally as any function that makes a comparison between two cases, while a true metric is a metric in a mathematical sense. This means that a true metric is a function $d$ that satisfy the following three axioms where $X$ denotes the case base with the set of all cases:

1. $d(\boldsymbol{x}, \boldsymbol{y}) \geq 0$ (non-negative and identity) with $d(\boldsymbol{x}, \boldsymbol{y}) = 0$ if and only if $\boldsymbol{x} = \boldsymbol{y}$, for all $\boldsymbol{x}, \boldsymbol{y} \in X$
2. $d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$ (symmetric) for all $x, y \in X$
3. $d(\boldsymbol{x}, \boldsymbol{z}) \leq d(\boldsymbol{x}, \boldsymbol{y}) + d(\boldsymbol{y}, \boldsymbol{z})$ (triangle inequality) for all $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in X$

There is a discussion in the CBR literature whether all of the above axioms are required for useful similarity and distance metrics [28, 29]. Common true metrics that we will use in this paper is the Manhattan distance and the Euclidean distance. The definition of the Manhattan distance is

$$d(\boldsymbol{x}, \boldsymbol{y}) = \sum_k |\boldsymbol{x}^k - \boldsymbol{y}^k|$$

where $|\ldots|$ denotes the absolute value function and $k$ denotes a case attribute. The Euclidean distance is defined as

$$d(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_k |\boldsymbol{x}^k - \boldsymbol{y}^k|^2}$$

### 3.2 Statistical Metrics

A commonly used statistical metric for comparing two probability distributions is the Kullback-Leibler divergence (KL) [30]. KL is also sometimes called the relative entropy or the information gain, since it is closely related to the entropy concept introduced by Shannon [31, 32].

The KL for the two probability distributions $p_i, p_j$, with parameter $\theta$, that are two probability density functions for continuous variables and probability mass functions in case of discrete variables:

$$D(p_i \| p_j) = \int \log\left(\frac{p_i(\theta)}{p_j(\theta)}\right) p_i(\theta) d\theta \qquad (1)$$

In case of discrete parameters, the integral is replaced with a sum.

KL is not symmetric but it can be made symmetric by computing the KL divergence in both directions and then add them together. This is an important characteristic if we desire a true metric as described in Sect. 3.1. The symmetric KL is often called Jeffreys divergence (J-divergence). The J-divergence will then be:

$$\begin{aligned} J(p_i, p_j) &= D(p_i \| p_j) + D(p_j \| p_i) \\ &= \int \log\left(\frac{p_i(\theta)}{p_j(\theta)}\right) p_i(\theta) d\theta + \int \log\left(\frac{p_j(\theta)}{p_i(\theta)}\right) p_j(\theta) d\theta \\ &= \int \log\left(\frac{p_i(\theta)}{p_j(\theta)}\right) \big(p_i(\theta) - p_j(\theta)\big) d\theta \end{aligned} \tag{2}$$

In this paper, we use the J-divergence as basis for the similarity metrics, because it is a commonly used measure and it has a clear information theoretical interpretation. Other statistical metrics for comparing distributions are also available such as the total variation distance, the Euclidean distance and the Jensen-Shannon divergence [33, 30, 34–36]. Later, we will see that there are connections between the J-divergence and other types of distances between probability distributions.

### 3.3 Logistic regression

Logistic regression is a binary classifier that can be considered a discrete version of linear regression [37]. Thus, it is a linear classifier that can only separate between classes that are linearly separable. However, since no assumption is made of the distribution of the independent variables it is less restrictive than the related naive Bayes classifier [38]. Assuming a binary classification with $c \in \{0, 1\}$ and feature vector $\boldsymbol{x}$, for logistic regression, we have:

$$\begin{aligned} p(c = 1|\boldsymbol{x}) &= \frac{1}{1 + \exp(-\boldsymbol{\omega}^T \boldsymbol{x})} \\ p(c = 0|\boldsymbol{x}) &= \frac{\exp(-\boldsymbol{\omega}^T \boldsymbol{x})}{1 + \exp(-\boldsymbol{\omega}^T \boldsymbol{x})} \end{aligned} \tag{3}$$

where $\boldsymbol{\omega}$ is a weight vector with $K+1$ weights assuming that $\boldsymbol{x}$ has $K+1$ features including an extra feature that is 1 for all cases.

Logistic regression can be generalized to the multiclass situation, called multinomial logistic regression, by training one classifier for each class – using one class against all other classes - and then combine the classifiers' predictions. The probability of a class $z \in \{1, 2, \ldots, m\}$ is computed as follows:

$$p(c = z|\boldsymbol{x}) = \frac{\exp(\boldsymbol{\omega}_z^T \boldsymbol{x})}{\sum_{z'=1}^{m} \exp(\boldsymbol{\omega}_{z'}^T \boldsymbol{x})} \tag{4}$$

where $\boldsymbol{\omega_z}$ is the fitted weight vector for each classifier and $m$ is the number of classes. Then, a new case is classified with the most probable class.

# 4 The Case-based Explanation Framework

This section applies the generic case-based explanation framework presented in [7] to classification. The framework justifies the predictions by estimating the system reliability case by case. By only considering probabilistic methods, we can give the framework a good theoretical foundation, while the explanation part can in principle be used for any classifier algorithm. The proposed approach for explaining classification is as follows:

1. *Classify a new case using the probabilistic model*
2. *Retrieve most similar previous cases using the defined similarity metric*
3. *For each previous case, classify the previous case using the probability model*
4. *Compute the local accuracy for the new case as the fraction of correctly classified previous cases*
5. *Present predicted class and the local accuracy together with most similar cases to the user*

Sect. 5 presents an application of this framework to a real example where we explain the predictions from a logistic regression model for diagnosing faults. However, before that, we will describe a generic approach to defining similarity metrics and derive metrics for comparing cases with respect to probabilistic classification.

## 4.1 Statistical Measures of Similarity for a Probabilistic Classifier

In this section, we present the principled approach to defining similarity metrics from probability distributions that was introduced in [7], and we apply it to probabilistic classification. We start by defining a basis for comparing two cases from a case base called *the principle of interchangeability*, and then, we derive four possible metrics. The principle of interchangeability is defined as follows:

**Definition 1.** *Two cases $\boldsymbol{x}_i, \boldsymbol{x}_j$ in case base $X$ are similar if they can be interchanged such that the two probability distributions $P_i$, $P_j$ inferred from excluding $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ respectively from the case base – $X \setminus \boldsymbol{x}_i$ and $X \setminus \boldsymbol{x}_j$ – are identical with respect to some parameter(s) of interest.*

As starting point for deriving similarity metrics, we use the discrete version of J-divergence from Sect. 3.2. The J-divergence between two cases $(c_i, \boldsymbol{x}_i), (c_j, \boldsymbol{x}_j)$ in case base $X$, with respect to the class distributions, is then:

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = J(p_i, p_j) = \sum_c \log \frac{p_i(c|\boldsymbol{x}_i)}{p_j(c|\boldsymbol{x}_j)} (p_i(c|\boldsymbol{x}_i)) - p_j(c|\boldsymbol{x}_j))) \qquad (5)$$

where $c$ is the class parameter and $p_i(c|\boldsymbol{x}_i)$ and $p_j(c|\boldsymbol{x}_j)$ are probability distributions of the class derived from the case base when excluding the cases $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ respectively.

The resulting measure between two cases can then be interpreted information theoretically as the sum of the information gain from including one over the other case and the information gain from including the other case over the first case in the case base. However, the resulting J-divergence distance is not a true metric, so an additional step might be needed that turns it into a final distance that fulfills the axioms of a true metric.

The Eq. 5 violates axiom 1 and axiom 3 of a true metric. However, by rewriting the Eq. 5, we can derive a lower and an upper limit that both are easily transformed into true metrics with respect to the class probability space and the class log-probability space respectively as follows

$$J(p_i, p_j) = \sum_c \big|\log(p_i(c|\boldsymbol{x}_i)) - \log(p_j(c|\boldsymbol{x}_j))\big|\big|p_i(c|\boldsymbol{x}_i) - p_j(c|\boldsymbol{x}_j)\big|$$

Then, since $|x-y| \leq |\log(x) - \log(y)|$ and $\max(|x-y|) = 1$ for all $x, y \in (0,1]$ we have:

$$\sum_c \big|p_i(c|\boldsymbol{x}_i) - p_j(c|\boldsymbol{x}_j)\big|^2 \leq J(P_i, P_j) \leq \sum_c \big|\log(p_i(c|\boldsymbol{x}_i)) - \log(p_j(c|\boldsymbol{x}_j))\big| \tag{6}$$

So the J-divergence is greater than or equal to the square of the Euclidean distance in the probability space and lesser than or equal to the Manhattan distance in the log-probability space. Thus, the upper and lower limits results in two more possible distances that we can use.

Notice that the metric in Eq. 5 assumes that the true classes for both cases are known, but in this paper the goal is to predict the class of a new case. So, assuming that $c_i$ is unknown for $\boldsymbol{x}_i$, we cannot estimate $p_j$ since $(c_i, \boldsymbol{x}_i)$ cannot be included in the case base, and thereby, we cannot compute the J-divergence exactly. Yet, if we have a large case base, then $p_i$ and $p_j$ would anyway be approximately equal, and hence, this would not be a problem. So, with a large enough case base, we can approximate $p_j \approx p_i$.

If we do not want to approximate $p_i$ and $p_j$ as equal, we either have to compute the J-divergence analytically, which might not be easy, or estimate the probability distributions for each case in the case base, which might be computationally heavy. Another, more pragmatic approach for managing this problem is to consider that when the class is known for case $\boldsymbol{x}_j$, we can actually model that as $p_j(c_j|\boldsymbol{x}_j) = 1$ and $p_j(c|\boldsymbol{x}_j) = 0$ for all other classes $c$. But, for a new case $\boldsymbol{x}_i$ with an unknown class, we estimate $p_i(c_i|\boldsymbol{x}_i)$ using a probabilistic model from all known cases. Then, by assuming that $p_i(c|\boldsymbol{x}_i) > 0$ for all classes $c$, we have the following:

$$\begin{aligned} J(p_i, p_j) =& \sum_c \log\left(\frac{p_i(c|\boldsymbol{x}_i)}{p_j(c|\boldsymbol{x}_j)}\right)(p_i(c|\boldsymbol{x}_i) - p_j(c|\boldsymbol{x}_j)) \\ =& \log\left(\frac{p_i(c_j|\boldsymbol{x}_i)}{1}\right)(p_i(c_j|\boldsymbol{x}_i) - 1) + \sum_{c \neq c_j} \log\left(\frac{p_i(c_j|\boldsymbol{x}_i)}{0}\right)(p_i(c|\boldsymbol{x}_i) - 0) \\ =& \log(p_i(c_j|\boldsymbol{x}_i))(p_i(c_j|\boldsymbol{x}_i) - 1) \ + \ \infty \end{aligned}$$

Since we are only interested in comparing distances relatively each other to find the closest cases, we can choose to ignore the infinity part and only compare the first term. Thus, we will get yet an alternative distance between two cases as follows:

$$d'(\boldsymbol{x}_i, \boldsymbol{x}_j) = \log\left(p_i(c_j|\boldsymbol{x}_i)\right)\left(p_i(c_j|\boldsymbol{x}_i) - 1\right) \tag{7}$$

However, this leads to a different definition of similarity, since we include $(c_j, \boldsymbol{x}_j)$ in the case base for estimating both $p_i, p_j$, and that we use two different distributions conditioned on whether the class is known.

We have now derived four different distances for comparing the similarity between cases relating to our definition of similarity and the J-divergence. The four derived distances are listed in Table 1. From now, we assume a large case base so that the distributions $p_i$ and $p_j$ are approximately equal.

**Table 1.** The four derived distances and two standard distances.

| Name | Distance | From Equation |
|---|---|---|
| J-divergence | $\sum_c \log \frac{p_i(c \mid \boldsymbol{x}_i)}{p_j(c \mid \boldsymbol{x}_j)} \left(p_i(c \mid \boldsymbol{x}_i) - p_j(c \mid \boldsymbol{x}_j)\right)$ | Eq. 5 |
| Approximate Prob | $\sqrt{\sum_c \left\lvert p_i(c \mid \boldsymbol{x}_i) - p_j(c \mid \boldsymbol{x}_j)\right\rvert^2}$ | Eq. 6 |
| Approximate Log-Prob | $\sum_c \left\lvert \log(p_i(c \mid \boldsymbol{x}_i)) - \log(p_j(c \mid \boldsymbol{x}_j)\right\rvert$ | Eq. 6 |
| Pragmatic | $\log\left(p_i(c_j \mid \boldsymbol{x}_i)\right)\left(p_i(c_j \mid \boldsymbol{x}_i) - 1\right)$ | Eq. 7 |

## 5 Explaining Fault Diagnosis

We will in this section apply the proposed approach for explaining fault diagnosis.

In fault diagnosis, faults are classified so that correct actions can be taken in order to minimize the cost of faults. Preferable, faults should be detected and classified before they harm the system. A measure of the confidence of a classification is also desirable so that no unnecessary actions are taken if the classification is wrong. Assuming that the probability model is correct, a high probability means a high confidence, while a low probability means a low confidence in the predicted value. However, if the probability model is locally error-prone, the probabilities cannot be trusted, as for instance, when the linearity assumption of logistic regression does not hold in the whole feature space. As one remedy, we propose to use the local accuracy as a complementing, and more intuitive, confidence measure. The local accuracy is computed locally, directly from the most similar cases, and thereby, it should be less affected by an error-prone probability model. Consequently, a high local accuracy should be able to justify a classification, while a low local accuracy should invalidate a classification. Thus, a user can use this approach to decide whether to trust a prediction case-by-case regardless of the global prediction accuracy of the algorithm.

In the following, given an error-prone logistic regression model (Sect. 5.1), we show that it is possible to train the $k$-nearest neighbor algorithm (kNN) to estimate the local accuracy (Sect. 5.2). Then, by looking at a case with inconsistent confidence measures – for instance, high probability and low local accuracy – we can detect bad prediction performance that is otherwise overlooked when only considering the class probability (Sect. 5.3). Last, we discuss how to interpret the local accuracy in other cases (Sect. 5.4).

For all experiments, we have used the implementation of logistic regression and kNN provided by the Scikit-learn Python module [39]. As example data set, we use the Steel plate faults data set [40] from the UC Irvine Machine Learning Repository [41]. The data set consists of more than 1900 cases with 7 types of steel plate faults and 27 different dependent variables listed in Table 2.

**Table 2.** Attributes: 27 independent variables, and the last rows shows the 7 fault classes.

| | | | |
|---|---|---|---|
| X_Minimum | X_Maximum | Y_Minimum | Y_Maximum |
| Pixels_Areas | X_Perimeter | Y_Perimeter | Sum_of_Luminosity |
| Min_of_Luminosity | Max_of_Luminosity | Length_of_Conveyer | TypeOfSteel_A300 |
| TypeOfSteel_A400 | Steel_Plate_Thickn. | Edges_Index | Empty_Index |
| Square_Index | Outside_X_Index | Edges_X_Index | Edges_Y_Index |
| Outside_Global_Index | LogOfAreas | Log_X_Index | Log_Y_Index |
| Orientation_Index | Luminosity_Index | SigmoidOfAreas | |
| 1. Pastry | 3. K_Scatch | 5. Dirtiness | 7. Other_Faults |
| 2. Z_Scratch | 4. Stains, | 6. Bumps, | |

### 5.1 Fitting Logistic Regression

In this section, we fit multinomial logistic regression to classify faults. As learning parameter, we use the $l1$-norm and the model regulation parameter is fine-tuned using grid search with 5-fold cross validation. Fig. 1 shows the learning curve of the overall accuracy for classifying faults. The learning curve was computed by splitting the data set 10 times into 70% training set and a 30% testing, and the results were then averaged. As can be seen, the training curve and validation curve are converging, but at a low level just above 0.7. Thus, more features or a more complex learning algorithm would be needed to improve on this. Hence, this is an error-prone probability model that we will use as an example to illustrate the proposed approach.

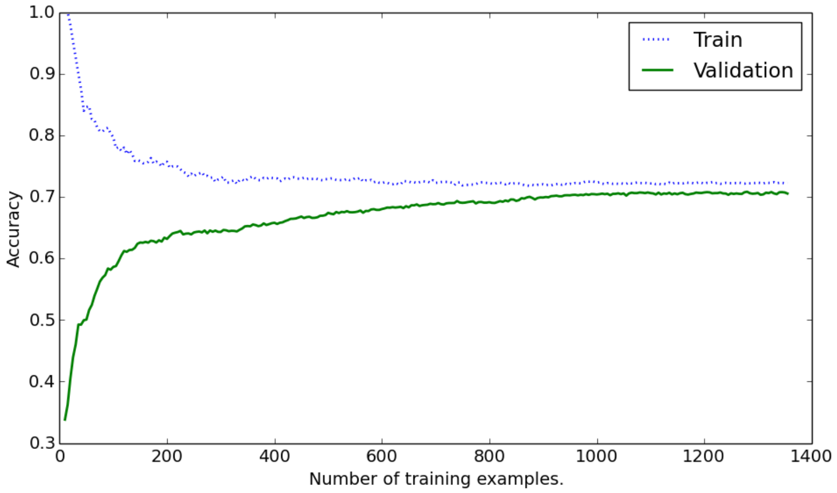

**Fig. 1.** The accuracy learning curve for classifying steel faults.

### 5.2 Estimating Local Accuracy with kNN

After fitting the multinomial logistic regression, we train the kNN algorithm to estimate the local accuracy. So, in this case, instead of predicting the class of a case, we use kNN to estimate a confidence in whether it will be classified correctly. Therefore, the classification label is replaced with 1 if a case in the training set was correctly classified and 0 otherwise. Then, kNN estimates the local accuracy by averaging the ones and zeros from the $k$-nearest neighbors, and then, the mean squared error (MSE) is used for evaluation:

$$mse(X) = \sum_{\boldsymbol{x} \in X} (C(\boldsymbol{x}) - A_k(\boldsymbol{x}))^2 \text{ with } A_k(\boldsymbol{x}) = \frac{\sum_{i=1}^{k} C(\boldsymbol{x}'_i)}{k} \tag{8}$$

where $X$ is a set of cases, $A(\boldsymbol{x}) \in [0,1]$ is the local accuracy for the $k$ most similar cases of $\boldsymbol{x}$ and $C(\boldsymbol{x}) = 1$ and $C(\boldsymbol{x}) = 0$ indicate correct and incorrect classification of $\boldsymbol{x}$ respectively. Then, if $A_k(\boldsymbol{x}) > 0.5$, it is more likely than unlikely that the classification is correct. This also means that the local accuracy can be interpreted as the probability of a correct classification.

For fine-tuning kNN, we split the data set 10 times into 60% training set, 20% validation set, and 20% testing set. Thereafter, we compute the average MSE over the validation and test sets. The distances in Table 1 were used together with the Manhattan and Euclidean distances in combination with the derived distances to weigh in the similarity between the cases directly. Notice that the class probabilities are considered as additional features. For all distances but J-divergence, we normalize so that each feature has a mean of 0 and a standard deviation of 1. The results are shown in Table 3 where Approximate Log-Prob distance has the lowest validation MSE. Fig. 2 plots the results for a varying number of $k$-neighbors.

In Fig 2, we notice that $k = 9$ is the best number of neighbors for Approaximate Log-Prob, but we must also consider how convincing it is to support a claim with only 9 neighbors. Thus, since there seems to be no larger differences between the MSE of $k \in [9, 15]$, we can at least use $k = 10$ as a more convincing explanation.

**Table 3.** The mean squared error (MSE) for different distances (best is in bold font).

| Distance | Validation MSE | Test MSE | k neighbors |
|---|---|---|---|
| Manhattan | 0.171 | 0.174 | 10 |
| Euclidean | 0.173 | 0.176 | 9 |
| Pragmatic | 0.214 | 0.215 | 60 |
| Approximate Prob | 0.175 | 0.173 | 17 |
| **Approximate Log-Prob** | **0.166** | **0.166** | **9** |
| Approximate Log-Prob Euclidean | 0.172 | 0.173 | 10 |
| Approximate Log-Prob Manhattan | 0.168 | 0.171 | 9 |
| J-Divergence | 0.176 | 0.173 | 19 |

### 5.3 Case-based Explanation Examples

Given a fitted logistic regression classifier and a fine-tuned kNN algorithm, we will now demonstrate the approach using two example faults. Table 4 and Table 5 show two examples where the target case is the fault that is being diagnosed.

**Table 4.** Fault 1: Low local accuracy 20% (2 of 10) and low class probability 33.4%.

| Attribute | Target | Case 1 | Case 2 | Case 3 | Case 4 | Case 5 |
|---|---|---|---|---|---|---|
| X_Minimum | 57.0 | 127.0 | 205.0 | 843.0 | 23.0 | 282.0 |
| ... | | | | | | |
| SigmoidOfAreas | 0.1753 | 0.2253 | 0.2359 | 0.215 | 0.2051 | 0.1954 |
| True Class | (6) | 7 | 6 | 7 | 7 | 6 |
| Predicted Class | 7 | 7 | 7 | 6 | 3 | 7 |
| Probability of Class | 0.334 | 0.503 | 0.5 | 0.476 | 0.444 | 0.364 |

For both faults, the estimated local accuracy is very low: only 1 or 2 out of 10 of the most similar cases are correctly classified. However, for the first fault, the probability of the predicted class is also low, and thus, it is consistent with the local accuracy. So, we have no reason to believe that the estimated probability is wrong. In contrast, for the second fault, the probability of the predicted class

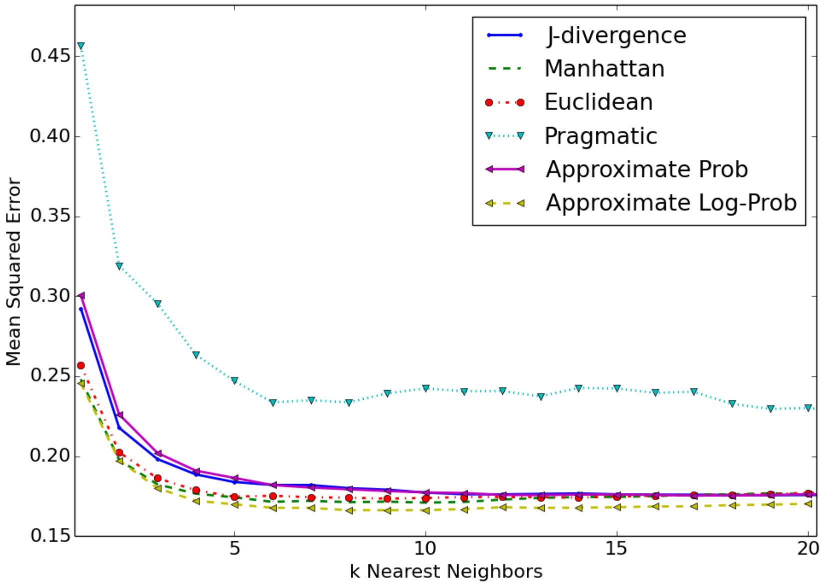

**Fig. 2.** MSE for the kNN algorithm using different distances and various $k$.

is quite high, despite that the local accuracy is very low. Then, if we also look at the probability of the predicted class for the five most similar cases shown in Table 5, it is quite high for all of them, although the wrong class is predicted in each case. So, we have good reason to doubt the prediction of the classifier, even though the classifier is quite confident in being right.

**Table 5.** Fault 2: Low local accuracy 10% (1 of 10) and high class probability 73.4%.

| Attribute | Target | Case 1 | Case 2 | Case 3 | Case 4 | Case 5 |
|---|---|---|---|---|---|---|
| X_Minimum | 57.0 | 61.0 | 623.0 | 10.0 | 1067.0 | 1338.0 |
| ... | | | | | | |
| SigmoidOfAreas | 0.1753 | 0.1659 | 0.2018 | 0.1753 | 0.215 | 0.2195 |
| True Class | (6) | 6 | 1 | 6 | 1 | 6 |
| Predicted Class | 7 | 7 | 7 | 7 | 7 | 7 |
| Probability of Class | 0.734 | 0.663 | 0.636 | 0.826 | 0.493 | 0.86 |

### 5.4 Analyzing the Local Accuracy

In the previous section, we saw two examples where in the first example the proposed approach justifies the probability model's prediction but invalidates the second example. Thus, we have shown that the proposed approach is able to detect when the probability model does not perform well. However, two more interesting situations to compare are when the prediction has high probability with a high local accuracy and when the prediction has low probability with a high local accuracy. In both cases, the local accuracy indicates that the prediction can be trusted, although the second case has low probability. In the latter case, this mean that the probability model does not make a good probability estimation but that the model anyway makes a good prediction of the most probable class. Clearly, the problem of deciding when to trust a prediction can be viewed as a decision theoretical problem and the solution will likely depend on the application domain, for instance, the cost of a misclassification. However, due to brevity, we will not further deal with the problem of how to decide whether the local accuracy is high or low, or what to do when it is neither high nor low, but leave that to future work.

## 6 Conclusions and Future Work

In this paper, we have extended the framework for knowledge light case-based explanation proposed in [7] to probabilistic classification, and we have applied it for explaining fault diagnosis. Thus, a major contribution of this work is a principled and theoretically well-founded approach to defining similarity metrics for retrieving cases relative to a probability model for classification.

A second contribution is a novel approach to justifying a prediction by computing the local accuracy as the fraction of the most similar cases that are classified correctly. Since the justification is based on real cases and not merely on the correctness of the probability model, we argue that this is a more intuitive justification of the reliability than only considering the estimated probability as a measure of confidence. For instance, as noted in Sect. 5.3, with this approach, the users can easily detect when the prediction uncertainty of the probability model does not agree with the computed local accuracy, and therefore, judge for themselves whether to trust a prediction or not. Since the accuracy is computed locally, the proposed approach addresses the problem that a probability model might not perform consistently over the whole feature space.

An interesting future development – as already noted in previous section – is to further investigate the use of the local accuracy for deciding when to trust or not trust a prediction. In addition, this approach can also be used for selecting which classifier to use for a new case. Another future research direction is to develop CBR applications where the main task is a case-based prediction and not just as a complement to a probabilistic prediction.

**Acknowledgments** The authors gratefully acknowledge the funding from the Swedish Knowledge Foundation (KK-stiftelsen) [42] through ITS-EASY Research School, the European ARTEMIS project ME3Gas (JU Grant Agreement number 100266), the ITEA 2 (ITEA 2 Call 5, 10020) and Swedish Governmental Agency for Innovation Systems (VINNOVA) grant no 10020 and JU grant no 100266 making this research possible.

## References

1. Wick, M.R., Thompson, W.B.: Reconstructive expert system explanation. Artificial Intelligence **54**(1) (1992) 33–70
2. Ye, L.R., Johnson, P.E.: The impact of explanation facilities on user acceptance of expert systems advice. Mis Quarterly (1995) 157–172
3. Gregor, S., Benbasat, I.: Explanations from intelligent systems: Theoretical foundations and implications for practice. MIS quarterly (1999) 497–530
4. Leake, D., McSherry, D.: Introduction to the special issue on explanation in case-based reasoning. Artificial Intelligence Review **24**(2) (2005) 103–108
5. Darlington, K.: Aspects of intelligent systems explanation. Universal Journal of Control and Automation **1** (2013) 40 – 51
6. Langlotz, C.P., Shortliffe, E.H.: Adapting a consultation system to critique user plans. International Journal of Man-Machine Studies **19**(5) (1983) 479–496
7. Olsson, T., Gillblad, D., Funk, P., Xiong, N.: Case-based reasoning for explaining probabilistic machine learning. International journal of computer science & information technology (IJCSIT) **6**(2) (April 2014)
8. Isermann, R.: Supervision, fault-detection and fault-diagnosis methods–an introduction. Control engineering practice **5**(5) (1997) 639–652
9. Jayaswal, P., Wadhwani, A., Mulchandani, K.: Machine fault signature analysis. International Journal of Rotating Machinery (2008)
10. Isermann, R.: Fault-diagnosis systems: an introduction from fault detection to fault tolerance. Springer Verlag (2006)
11. Olsson, E.: Fault Diagnosis of Industrial Machines Using Sensor Signals and Case-Based Reasoning. School of Innovation, Design and Engineering, Mälardalen University (2009)
12. Olsson, T., Funk, P.: Case-based reasoning combined with statistics for diagnostics and prognosis. Journal of Physics: Conference Series **364**(1) (2012) 012061
13. Caruana, R., Kangarloo, H., Dionisio, J., Sinha, U., Johnson, D.: Case-based explanation of non-case-based learning methods. In: Proceedings of the AMIA Symposium, American Medical Informatics Association (1999) 212
14. Nugent, C., Cunningham, P.: A case-based explanation system for black-box systems. Artificial Intelligence Review **24**(2) (2005) 163–178
15. Schank, R.C., Leake, D.B.: Creativity and learning in a case-based explainer. Artificial Intelligence **40**(1) (1989) 353–385
16. Aamodt, A.: Explanation-driven case-based reasoning. In: Topics in case-based reasoning. Springer (1994) 274–288
17. Doyle, D., Tsymbal, A., Cunningham, P.: A review of explanation and explanation in case-based reasoning. Dublin, Trinity college https://www. cs. tcd. ie/publications/tech-reports/reports **3** (2003)
18. Cunningham, P., Doyle, D., Loughrey, J.: An evaluation of the usefulness of case-based explanation. In: Case-Based Reasoning Research and Development. Springer (2003) 122–130
19. McSherry, D.: Explanation in case-based reasoning: an evidential approach. In: Proceedings of the 8th UK Workshop on Case-Based Reasoning. (2003) 47–55
20. McSherry, D.: Explaining the pros and cons of conclusions in cbr. In: Advances in Case-Based Reasoning. Springer (2004) 317–330
21. McSherry, D.: A lazy learning approach to explaining case-based reasoning solutions. In: Case-Based Reasoning Research and Development. Springer (2012) 241–254

22. Doyle, D., Cunningham, P., Bridge, D., Rahman, Y.: Explanation oriented retrieval. In: Advances in Case-Based Reasoning. Springer (2004) 157–168
23. Cummins, L., Bridge, D.: Kleor: A knowledge lite approach to explanation oriented retrieval. Computing and Informatics **25**(2-3) (2006) 173–193
24. Nugent, C., Cunningham, P., Doyle, D.: The best way to instil confidence is by being right. In: Case-Based Reasoning Research and Development. Springer (2005) 368–381
25. Wall, R., Cunningham, P., Walsh, P.: Explaining predictions from a neural network ensemble one at a time. In: Principles of Data Mining and Knowledge Discovery. Springer (2002) 449–460
26. Green, M., Ekelund, U., Edenbrandt, L., Björk, J., Hansen, J., Ohlsson, M.: Explaining artificial neural network ensembles: A case study with electrocardiograms from chest pain patients. In: Proceedings of the ICML/UAI/COLT 2008 Workshop on Machine Learning for Health-Care Applications. (2008)
27. Green, M., Ekelund, U., Edenbrandt, L., Björk, J., Forberg, J.L., Ohlsson, M.: Exploring new possibilities for case-based explanation of artificial neural network ensembles. Neural Networks **22**(1) (2009) 75–81
28. Burkhard, H.D., Richter, M.M.: On the notion of similarity in case based reasoning and fuzzy theory. In: Soft computing in case based reasoning. Springer (2001) 29–45
29. Burkhard, H.D.: Similarity and distance in case based reasoning. Fundamenta Informaticae **47**(3) (2001) 201 – 215
30. Kullback, S., Leibler, R.A.: On information and sufficiency. The Annals of Mathematical Statistics **22**(1) (1951) 79–86
31. Ihara, S.: Information theory for continuous systems. Volume 2. World Scientific (1993)
32. Shannon, C.E.: A mathematical theory of communication. ACM SIGMOBILE Mobile Computing and Communications Review **5**(1) (2001) 3–55
33. Rachev, S.T., Stoyanov, S.V., Fabozzi, F.J., et al.: A probability metrics approach to financial risk measures. Wiley. com (2011)
34. Lin, J.: Divergence measures based on the shannon entropy. Information Theory, IEEE Transactions on **37**(1) (1991) 145–151
35. Cha, S.H.: Comprehensive survey on distance/similarity measures between probability density functions. City **1**(2) (2007) 1
36. Dragomir, S.C.: Some properties for the exponential of the kullback-leibler divergence. Tamsui Oxford Journal of Mathematical Sciences **24**(2) (2008) 141–151
37. Murphy, K.P.: Machine learning: a probabilistic perspective. MIT Press (2012)
38. Ng, A.Y., Jordan, M.I.: On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. Advances in neural information processing systems **2** (2002) 841–848
39. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12** (2011) 2825–2830
40. Steel Plates Faults Data Set. Source: Semeion, Research Center of Sciences of Communication, Via Sersale 117, 00128, Rome, Italy. www.semeion.it. https://archive.ics.uci.edu/ml/datasets/Steel+Plates+Faults (Last accessed: May 2014)
41. Bache, K., Lichman, M.: UCI machine learning repository (2013)
42. KK-Stiftelse: Swedish Knowledge Foundation. http://www.kks.se (Last Accessed: September 2013)