

Chapter 8

Explaining the Predictions of an Arbitrary Prediction Model: Feature Contributions and Quasi-nomograms



Erik Štrumbelj and Igor Kononenko

Abstract Acquisition of knowledge from data is the quintessential task of machine learning. The knowledge we extract this way might not be suitable for immediate use and one or more data postprocessing methods could be applied as well. Data postprocessing includes the integration, filtering, evaluation, and explanation of acquired knowledge. Nomograms, graphical devices for approximate calculations of functions, are a useful tool for visualising and comparing prediction models. It is well known that any generalised additive model can be represented by a quasi-nomogram – a nomogram where some summation performed by the human is required. Nomograms of this type are widely used, especially in medical prognostics. Methods for constructing such a nomogram were developed for specific types of prediction models thus assuming that the structure of the model is known. In this chapter we extend our previous work on a general method for explaining arbitrary prediction models (classification or regression) to a general methodology for constructing a quasi-nomogram for a black-box prediction model. We show that for an additive model, such a quasi-nomogram is equivalent to the one we would construct if the structure of the model was known.

8.1 Introduction

The field of nomography was invented at the end of the 19th century by Maurice d'Ocagne [8]. Up to the final quarter of the 20th century, nomograms were widely used as graphical computers for tasks such as navigation, projectile trajectories, and

E. Štrumbelj (✉) · I. Kononenko
Faculty of Computer and Information Science, University of Ljubljana,
Večna pot 113, 1000 Ljubljana, Slovenia
e-mail: erik.strumbelj@fri.uni-lj.si

I. Kononenko
e-mail: igor.kononenko@fri.uni-lj.si

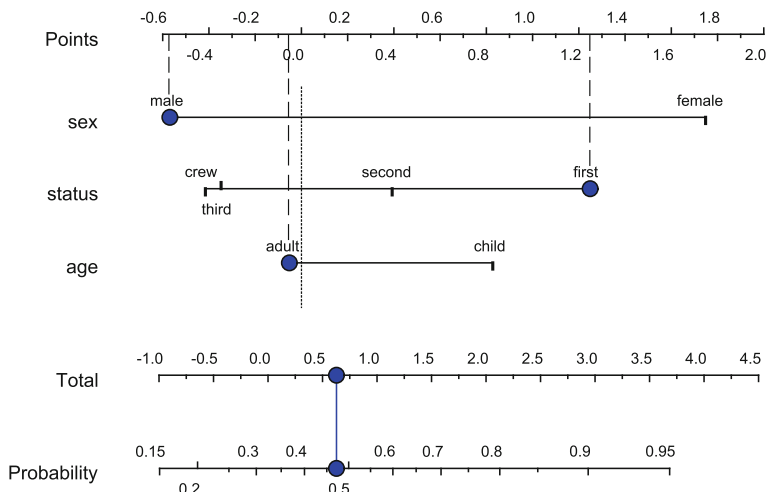


Fig. 8.1 We used the Orange data mining software [7] to produce this quasi-nomogram for the Naive Bayes model on the well known *Titanic* data set. A survival prediction for a specific instance, for example, an adult male travelling first class, is constructed in the following way. First, we mark the value of each input variable on its corresponding axis (three topmost dots) and read their points-contribution on the Points axis (vertical dashed lines). The summation of the three points-contributions is marked on the Total axis ($-0.6 - 0.05 + 1.25 \approx 0.6$). By drawing a straight vertical line, we can convert the sum into a probabilistic prediction. In this case, approximately 45%. Note that the range of the Points scale is determined by the minimum and maximum possible point contribution across all input variable values, while the range of the Total scale is determined by the minimum and maximum possible sum of point contributions across all input variables. The Points and Total axes need not be aligned, because the point summation has to be done manually

other tasks that require the computation of complex formulas. For more information see Doerfler’s survey of classical nomography [9].

In this chapter we focus on a specific type of graphical representation (see Fig. 8.1). Unlike a classical nomogram, Fig. 8.1 does not facilitate graphical-only computation (using a straightedge and a pencil), but requires additional summation. In recent years there has been a resurgence of interest in such graphical representations, especially in medical diagnostics and prognostics [6, 10, 15, 16, 24].¹ Note that nearly all related publications simply refer to such graphical representations as nomograms (sometimes as Kattan nomograms), although they do not completely fit the classical definition of a nomogram. In this chapter we promote a clear distinction between the two types of graphical representations and refer to Fig. 8.1 as a quasi-nomogram - a nomogram that requires some additional (non-graphical) computation.

Quasi-nomograms serve a dual purpose. First, they are a tool for producing “offline” predictions. And second, they provide the human with information about

¹www.sciencedirect.com currently lists 1393 research papers that feature the word “nomogram” in the title, keywords, or abstract and were published between 2006 and 2015. Most of them are from the medical field.

the model and the effect of the predictor variables on the target variable. As such, they are a useful tool for decision support and for providing non-experts, for example, patients in a hospital, with insight into their diagnosis and prognosis. Furthermore, quasi-nomograms can also be used by data mining practitioners as a model visualisation and inspection tool. As with other model-visualisation techniques, the informativeness and usefulness of the quasi-nomogram visualisation decreases as the number of input variables increases. However, in practical applications, such as medical prognostics, the number of input variables rarely exceeds 10.

Any generalised additive model can easily be visualised with a quasi-nomogram, which has motivated several model-specific approaches for constructing a quasi-nomogram. In this chapter, however, we show how a quasi-nomogram can be constructed for any (generalised) additive model in a uniform way and, with possible loss of prediction accuracy, for any prediction model. The main idea is to decompose the (unknown) model function into generalised additive form and then estimate each input variable's contribution function on a point-by-point basis. This is made possible by the fact that we do not need the analytical form of these functions to produce a quasi-nomogram. Instead, we only need to plot the functions in some finite resolution.

The remainder of the paper is as follows. The next Section describes our previous work and other related work in the broader area of explaining prediction models and the particular case of using nomograms to visualise models. In Sect. 8.3, we describe the construction of a nomogram for a black-box prediction model. Section 8.4 illustrates the use of the method on several data sets and models. Section 8.5 concludes the paper.

8.2 Explaining the Predictions with Feature Contributions

Prediction models are an integral part of knowledge discovery. How we choose a prediction model for a particular task strongly depends on the problem area. Sometimes we are primarily interested in prediction accuracy, on other occasions, interpretability is equally, if not more important.

Better interpretability is easily achieved by selecting a transparent model. However, more transparent models are usually less accurate than more complex models, often to an extent that we would rather consider making the latter more interpretable. As a result, numerous model-specific and general post-processing methods that provide additional explanation have been proposed (see [2, 22, 29, 35–37] and references therein).

Generating a nomogram for an arbitrary model is directly connected to developing a general (black-box) approach to explaining or increasing the interpretability of a prediction model. We illustrate this problem with a simple linear model $f(x_1, \dots, x_n) = f(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_n x_n$. If we standardise the input features, we could interpret the coefficient β_i as the i -th feature's global importance (in

statistical literature, variable importance). Note that using the features' global importance to select only a subset of features is analogous to a filter method for feature selection.

While the global importance reveals which features are more important, it does not tell us how features influence individual predictions. The difference between what the i -th feature contributes when its value is x_i and what it is expected to contribute:

$$\begin{aligned}\varphi_i(x) &= f(x) - E[f(x)|i\text{-th value unknown}] = \\ &= \beta_i x_i - \beta_i E[X_i],\end{aligned}\tag{8.1}$$

gives us such a local contribution. Equation (8.1) is also known as the situational importance of $X_i = x_i$ [1].

Because our model is additive, the local contribution of $X_i = x$ is the same across all instances where $X_i = x$, regardless of the values of other features.

The above illustrative example is simple, with a known model that is also additive. However, in our problem setting, we want a general method. The model has to be treated as a black-box - no assumptions are made other than that the model maps from some known input feature space to a known codomain. Therefore, we are limited to sensitivity analysis - changing the inputs and observing the outputs.

General (black-box) approaches are at a disadvantage - not being able to exploit model-specifics makes it more difficult to develop an effective and efficient method. However, being applicable to any type of model also has its advantages. It facilitates comparison of different types of models and, in practical applications, eliminates the need to replace the explanation method when the underlying model is changed or replaced.

Previously developed general approaches [21, 32, 39] tackle the problem in a similar one-feature-at-a-time way. That is, the contribution of a feature is

$$\varphi_i(x) = f(x_1, \dots, x_n) - E[f(x_1, \dots, X_i, \dots, x_n)]\tag{8.2}$$

Equation (8.2) is the difference between a prediction for an instance and the expected prediction for the same instance if the i -th feature is not known.

In practice, expression Eq. (8.2) can be efficiently approximated (or computed, if the feature's domain is finite). Additionally, if f is an additive model, Eq. (8.2) is equivalent to Eq. (8.1), so we do not lose any of the previously mentioned advantages associated with explaining an additive model.

However, when the features interact, as is often the case, the one-feature-at-a-time approach gives undesirable results. For example, observe the model $f(x_1, x_2) = x_1 \vee x_2$, where both features are uniformly distributed on $\{0, 1\}$. When computing the contribution of the first feature for $f(1, 1) = 1$, we see that perturbing its value does not change the prediction - the first feature's contribution is 0. The same holds for the second feature. Therefore, both features get a 0 contribution, which is clearly incorrect.

This example shows that perturbing one feature at a time gives undesirable results. All subsets have to be taken into account to avoid such issues.

In our previous work, we developed an alternative general approach that tackles the disadvantages of other general approaches described above [35–37]. To summarise the main ideas of the approach, let $\mathcal{X} = [0, 1]^n$ be our feature space and let $f : \mathcal{X} \rightarrow \mathfrak{R}$ represent the model that is used to predict the value of the target variable for an instance $x \in \mathcal{X}$. To avoid the shortcomings of other general methods, we observe the contribution of each subset of feature values. For this purpose, Eq. (8.2) is generalised to an arbitrary subset of features:

$$f_Q(x) = \mathbb{E}[f | X_i = x_i, \forall i \in Q], \quad (8.3)$$

where $Q \subseteq S = \{1, 2, \dots, n\}$ represents a subset of features. This allows us to define the contribution of a subset of feature values:

$$\Delta_Q(x) = f_Q(x) - f_{\emptyset}(x). \quad (8.4)$$

Equation (8.4) can be interpreted as the change in prediction caused by observing the values of a certain subset of features for some instance $x \in \mathcal{X}$.

To obtain each individual feature's local contribution, we map these 2^n terms onto n contributions, one for each feature's value. First, we implicitly define interactions by having the contribution of a subset of feature values equal the sum of all interactions across all subsets of those feature values:

$$\Delta_Q(x) = \sum_{W \subseteq Q} \mathcal{I}_W(x), \quad (8.5)$$

which, together with $\mathcal{I}_{\emptyset}(x) = 0$ (an empty set of features contributes nothing), uniquely defines the interactions:

$$\mathcal{I}_Q(x) = \Delta_Q(x) - \sum_{W \subset Q} \mathcal{I}_W(x). \quad (8.6)$$

Finally, each interaction is divided among the participating feature values, which defines the i -th features local contribution:

$$\varphi_i(x) = \sum_{W \subseteq S \setminus \{i\}} \frac{\mathcal{I}_{W \cup \{i\}}(x)}{|W| + 1}. \quad (8.7)$$

Figure 8.2 shows two example explanations for an instance from the *monks1* data set (binary class has value 1 iff the value of the 1st feature equals the 2nd feature or the 5th feature's value is 1). The Naive Bayes model, due to its assumptions of conditional independence of input features, cannot model the importance of the equivalence between *attr1* and *attr2*. Despite this limitation, it correctly predicts the class value, because for this instance, *attr5* = 1 is sufficient. The artificial neural network correctly models both concepts.

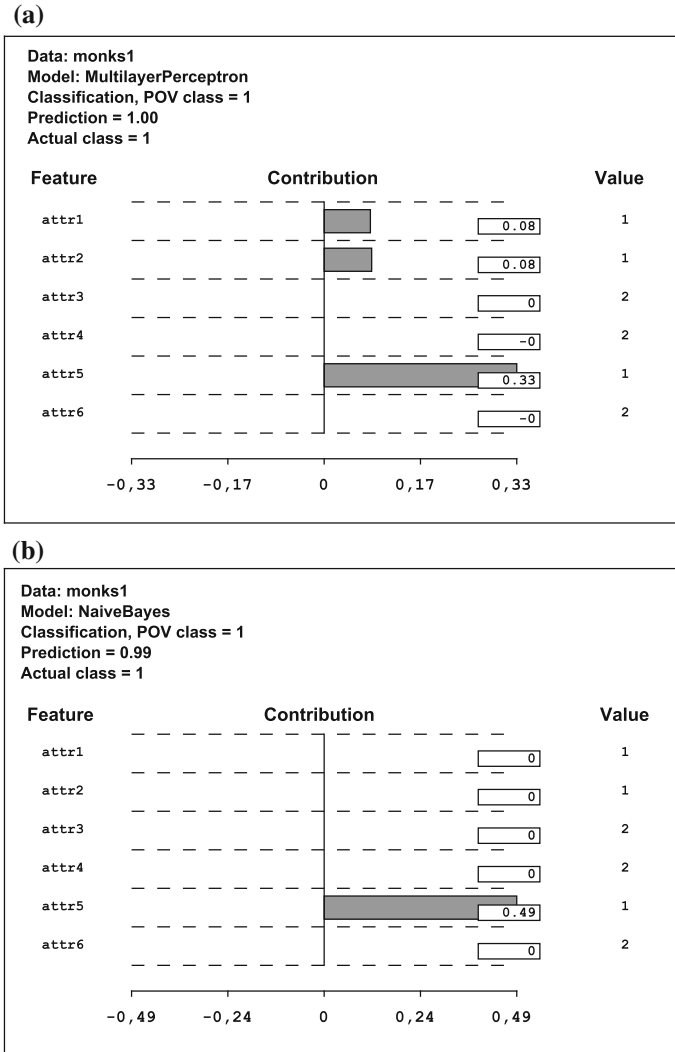


Fig. 8.2 Two visualisations of local contributions for two different models and the same instance from the *monks1* data set. The top of each visualisation shows information about the data set, model, prediction and the actual class value for this instance. The features' names and values for this instance are on the right- and left-hand side, respectively. The value of each feature's local contributions is shown in the corresponding box and visualised

The proposed method correctly reveals which features contribute. On the other hand, one-feature-at-a-time approaches would assign a 0 contribution to all features in the artificial neural network case. Perturbing just one feature does not change the model's prediction.

Equation (8.7) is shown to be equivalent to the Shapley value [33] of a coalitional game, where features are considered players and the generalised contribution is the worth of a coalition of features. This implies several desirable properties (see [35] for a more detailed and formal treatment):

- the local contributions sum up to the difference between the prediction and the expected prediction if no features are known,
- if two features have a symmetrical role in the model, they get the same contribution, and
- if a feature has no role, its contribution is 0.

This also facilitates a game-theoretic interpretation of the shortcoming of existing methods. By not correctly taking into account all interactions they violate the first property and can divide among features more (or less) than what their total worth is.

Computing the proposed local contribution has an exponential time complexity, which limits its practical usefulness. We use an alternative formulation of the Shapley value to derive an efficient sampling-based approximation, which allows us to compute the contributions in polynomial time [35]. The algorithm is extended with an online estimation of the approximation error. This provides a flexible mechanism for trade-off between running times and approximation error. The approximation algorithm is also highly parallelisable - it can be parallelised down to computing a single prediction.

We also considered two improvements that reduce running times (or approximation errors). First, the use of low-discrepancy or quasi-random sequences can improve the convergence of Monte Carlo integration [13]. We used Sobol quasi-random sequences. And second, not all features are equally important and, intuitively, less important features require fewer samples. We derive the optimal way of distributing a finite number of samples between features to minimise the expected approximation error across all n contributions. Empirical results show that non-uniform sampling substantially improves convergence, while quasi-random sampling results in a relatively small improvement. Note that the overall time complexity of generating our nomogram is $O(c \cdot n \cdot M(n))$, where n is the number of input variables, $M(n)$ is the time complexity of generating a single prediction (depends on the model), and c is a number-of-samples-per-feature constant that depends on the desired error and resolution of the nomogram lines (or number of distinct values, for features with a finite number of unique values), but not on the number of input variables n (that is, it does not increase with the number of features).

Generalised additive models are, by definition, written as a sum of the effects of individual input variables, transformed by some link function. Therefore, it is relatively straightforward to visualise any generalised additive model with a quasi-nomogram (see Sect. 8.3 for details). This has led to several model-specific methods for explaining several different types of statistical and machine learning models typically used for prediction: Support Vector Machines [5, 14, 38], the Naive Bayes classifier [20, 23], logistic regression [40]. The proposed approach, however, decomposes an individual prediction among features in a way that takes into account the other features' values. For each feature and its value, we can compute the mean local

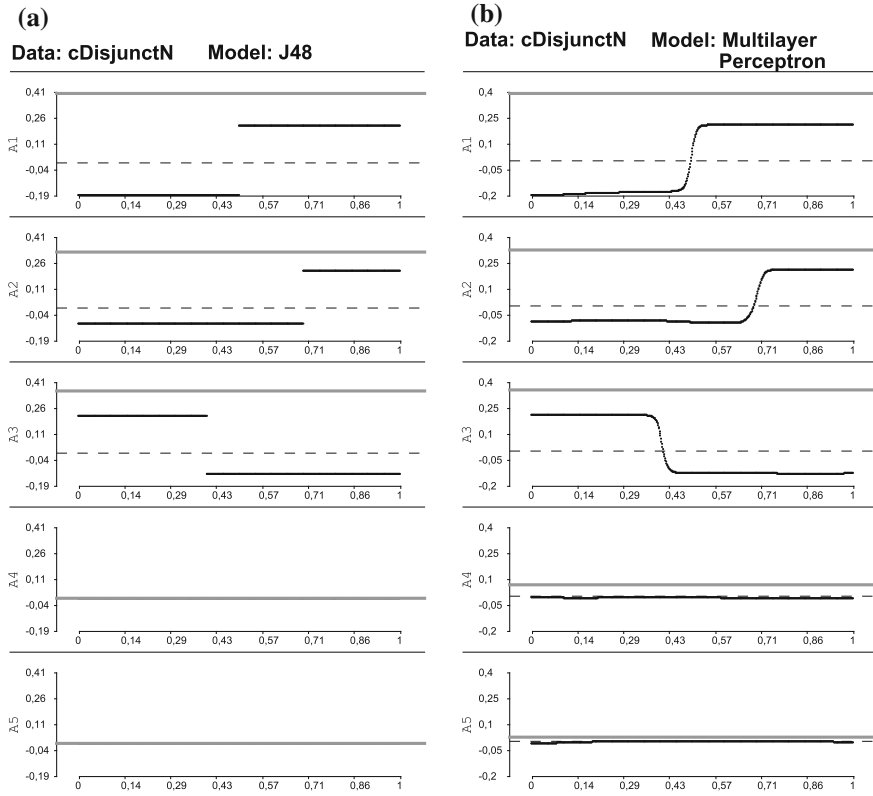


Fig. 8.3 Both models learn the concepts behind the data and the plotted average contribution functions (black) reveal where the individual features' contribution changes from negative to positive. The grey horizontal line represents the root feature importance of the feature

contribution when the feature has that value [36]. This produces, for each feature, a marginal effect function, which is similar to the marginal effect functions used in the construction of nomograms. In fact, we can show that if the underlying model is additive, this will produce equivalent results and this will be the basis for our construction of a nomogram for an arbitrary prediction model (see Sect. 8.3).

Figure 8.3 shows a pair of marginal effect visualisations for two different models on the same *cDisjunctN* data set. Out of the five features, only the first three are relevant. The class value equals 1 if (and only if) $A_1 > 0.5$ or $A_2 > 0.7$ or $A_3 < 0.4$. The visualisation reveals the difference between the step-function fit of the decision tree and smooth fit of the artificial neural network. It also reveals that the artificial neural network slightly overfits the two irrelevant features.

A feature's importance - the variance of its local contributions - can also be efficiently computed using a similar sampling-based approach. In combination with any learning algorithm the global importance can be used as a filter for feature

selection. This approach is similar to the in-built feature importance of Random Forests [4] and related to the LMG variable importance method for linear regression models [11].

Note that the practical advantages of explaining model predictions with feature contributions have been established with two applications. Firstly, an application to breast cancer recurrence predictions [34], where it was shown that in 95% of the cases the oncologist agreed with both the direction and the magnitude of the contribution. Furthermore, oncologists found the explanations a beneficiary tool and helpful in increasing their trust in the model's predictions. And second, a survey which showed that providing an explanation significantly improves the humans' predictions and also increases confidence [37]. The usefulness of such an explanation method as a tool for machine learning practitioners is further supported by several documented uses by other researchers in different areas. These include maximum shear stress prediction from hemodynamic simulations [3, 28], coronary artery disease diagnosis from medical images [18, 19], businesses' economic quality prediction [27] and the use of the explanation method to explain the PRBF classification network [30, 31].

8.3 Constructing the Quasi-nomogram

Take a response random variable Y and a set of predictor variables X_1, X_2, \dots, X_n . In a standard prediction setting, we are interested in how the response variable depends on the values of the predictor variables. We model this relationship with a model f , such that $f(x_1, x_2, \dots, x_n) = \mathbb{E}(Y)$. Usually, f is trained (inferred, estimated,...) using a set of labelled training instances $\{(x_{i,1}, x_{i,2}, \dots, x_{i,n}, y_i) \in [0, 1]^n \times [0, 1]\}_{i=1}^N$. Without loss of generality, we assumed that the predictor variables' domain ω is a n -dimensional unit cube.

Transparent models, such as the linear regression model $f(x_1, x_2, \dots, x_n) = \beta_0 + \beta_1 x_1 + \dots + \beta_n x_n$ are self-explanatory.² When dealing with less transparent models, we often employ explanation methods and techniques that make the model easier to understand and use. Quasi-nomograms are one such method. They make the model more transparent and can be used for the computation of the model's predictions.

Generalised additive models are a family of models that we can effectively represent with quasi-nomograms similar to one in Fig. 8.1. A generalised additive model can be written as

$$f(x_1, x_2, \dots, x_n) = F^{-1} \left(\sum_{i=1}^n f_i(x_i) + \beta_0 \right) = \mathbb{E}(Y), \quad (8.8)$$

where F is a smooth and bijective link function which relates the expectation to the predictor variables. That is, we try to fit the effect functions f_i , such that

²Linear regression is, of course, just a special case of generalised additive model with identity link function and linear effect functions

$$F(\mathbb{E}(Y)) = \sum_{i=1}^n f_i(x_i) + \beta_0.$$

Because a generalised additive model can be written as a sum of functions of individual predictor variables, we can plot each effect function f_i separately. To reconstruct a model's prediction for a particular instance, we read and sum the values of f_i , for each i (and β_0). Finally, we transform the sum with F^{-1} , which can be done with a simple conversion scale ("Total and Probability" in Fig. 8.1).

The described procedure is simple and effective, but assumes that the structure of the (generalised additive) model f is known. Now we describe a method that can be used to produce a nomogram for any prediction model f . Given a model f and a link function F , we define a set of functions

$$g_i(x) = \mathbb{E}(F(f(X_1, X_2, \dots, X_i = x, \dots, X_n)) - F(f(X_1, \dots, X_n))),$$

for each $i = 1..n$. The value $g_i(x)$ can be viewed as the expected change in the model's output if the i -th predictor variable is set to x . Observe the model

$$g(x_1, \dots, x_n) = F^{-1} \left(\sum_{i=1}^n g_i(x_i) + \mathbb{E}(F(f(X_1, \dots, X_n))) \right). \quad (8.9)$$

The model in Eq. (8.9) transforms model f into a generalised additive model, without assuming the structure of f . The following useful property can be shown.

Theorem 8.1 *If f is a generalised additive model and F the corresponding link function then $g(x_1, \dots, x_n) = f(x_1, \dots, x_n)$, for all $(x_1, \dots, x_n) \in X_1 \times \dots \times X_n$.*

Proof Taking into account the theorem's assumptions:

$$\begin{aligned} g(x_1, \dots, x_n) &= F^{-1} \left(\sum_{i=1}^n g_i(x_i) + \mathbb{E}(F(f(X_1, \dots, X_n))) \right) = \\ &= F^{-1} \left(\sum_{i=1}^n \mathbb{E}(f_i(x_i) - f_i(X_i)) + \sum_{i=1}^n \mathbb{E}(f_i(X_i)) + \beta_0 \right) = \\ &= F^{-1} \left(\sum_{i=1}^n f_i(x_i) - \sum_{i=1}^n \mathbb{E}(f_i(X_i)) + \sum_{i=1}^n \mathbb{E}(f_i(X_i)) + \beta_0 \right) = \\ &= F^{-1} \left(\sum_{i=1}^n f_i(x_i) + \beta_0 \right) = f(x_1, x_2, \dots, x_n) \end{aligned} \quad (8.10)$$

That is, the predictions obtained from g will be the same as the original models' predictions, conditional to f being an additive model or a generalised additive model with known link function F .

To compute the transformed model g , we require functions g_i and

$$\mathbb{E}(F(f(X_1, \dots, X_n))).$$

The latter $\mathbb{E}(F(f(X_1, \dots, X_n))) = \int_{\omega} F(f(\dots))dP$ can be efficiently approximated with simple Monte Carlo integration

$$\mathbb{E}(F(f(X_1, \dots, X_n))) \approx \frac{1}{M} \sum_{j=1}^M F(f(x_{j,1}, x_{j,2}, \dots, x_{j,n})),$$

where the realisations $x_{j,k}$ are obtained by generating a sequence of random samples (that is, instances) according to some distribution P of the input variables' domain ω . Each point $g_i(x)$ can be estimated in a similar way

$$g_i(x) \approx \frac{1}{M} \sum_{j=1}^M (F(f(x_{j,1}, \dots, x_{j,i} = x, \dots, x_{j,n})) - F(f(x_{j,1}, \dots, x_{j,n}))).$$

Theorem 8.1 holds for any probability distribution. Therefore, we can choose a distribution that is more convenient for sampling, such as a uniform distribution or a distribution where predictor variables X_i are independently distributed. Note that in general the estimation converges towards the actual value independently of the number of dimensions n . Furthermore, for faster convergence, quasi-random sampling can be used instead of pseudo-random sampling [25, 26]. In our experiments, we used the Sobol low-discrepancy quasi-random sequence [13].

The primary application of the proposed approach is to (generalised) additive models. However, in practice it can also be applied to a potentially non-additive model. In such cases, we are interested in how close the transformed model g is to f and how good the prediction accuracy of g is. The farther away g is from f the less useful the quasi-nomogram is in terms of providing insight into f . It is also possible that g is not close to f but has a better prediction accuracy. In such cases, we should consider using g instead of f , because it is both a better predictor and easier to interpret.

Given a set of N labelled instances, we used the root mean squared error to estimate the model's prediction accuracy

$$e_{g,y} = \sqrt{\frac{1}{N} \sum_{i=1}^N (f(x_{i,1}, x_{i,2}, \dots, x_{i,n}) - y_i)^2}$$

and the distance between the original model f and the transformed model g

$$e_{f,g} = \sqrt{\frac{1}{N} \sum_{i=1}^N (f(x_{i,1}, x_{i,2}, \dots, x_{i,n}) - g(x_{i,1}, x_{i,2}, \dots, x_{i,n}))^2}.$$

8.4 Illustrative Examples

We start with a simple toy data set with three input variables A_1 , A_2 , and A_3 with continuous domains $[0, 1]$ and uniformly distributed values. Let the relationship between the input variables and the target variable Y be linear: $Y = 0.5A_1 - 1.5A_2$. We generated 1000 instances from this data set at random, labelled each instance using the aforementioned linear relationship, and used the instances to train a *multilayer perceptron* artificial neural network model.

Let f be this multilayer perceptron model. The structure of f is unknown, but we can access its value for any point. We used the procedure described in Sect. 8.3 to generate the quasi-nomogram shown in Fig. 8.4a. The quasi-nomogram consists of three effect functions (one for each input variable) and a conversion scale. Each individual effect function graph is used to convert the input variable's value into a point-contribution. This is done by first drawing a straight vertical line that connects the input variable's value to the plotted effect function and then a horizontal line that connects this point to the vertical Points scale, where the points-contribution of this value can be read. The sum of all three variables' points is selected on the left-hand side of the conversion scale and the prediction is readily available to be read on the right-hand side of the conversion scale. Large dots and corresponding lines in Fig. 8.4a illustrate this procedure for the instance $(0.6, 0.8, 0.2)$.

The relationship between the procedure from the previous paragraph and the equations in Sect. 8.3 (Eq. (8.9) in particular) is as follows. Each input variables' effect function is plotted separately, one point at a time. The estimated value $g_i(x)$ corresponds to the value of the i -th input variable's effect function at x . Therefore, horizontal and vertical lines are used to obtain the effect functions' values. The summation part of the procedure produces the sum in Eq. (8.9). The values on the left-hand side of the conversion scale range from the minimum to the maximum possible sum of the effect functions. What remains is to add the expectation $\mathbb{E}(F(f(X_1, \dots, X_n)))$ and in the case of a non-identity link function F , apply the inverse of the link function F^{-1} . Because F is a bijective and smooth function, both operations are realised simultaneously and the mapped values are written on the right-hand of the conversion scale, where the human can read the final value. Note that the Total to Probability conversion scale from the nomograms generated in Orange (see Fig. 8.1) serves the same purpose as the conversion scale.

Notice that the quasi-nomogram in Fig. 8.4a is for the *multilayer perceptron* model, which is not necessarily additive. In this case the model was close to additive and the additive transformation in Fig. 8.4a was more accurate than the original model ($e_{f,y} = 0.013$, $e_{g,y} = 0.009$, $e_{f,g} = 0.007$). For comparison, *linear regression*, which is well-suited for the linear problem, produces the following results ($e_{f,y} \approx e_{g,y} = 5.8 \times 10^{-11}$). With an additive model such as the linear regression

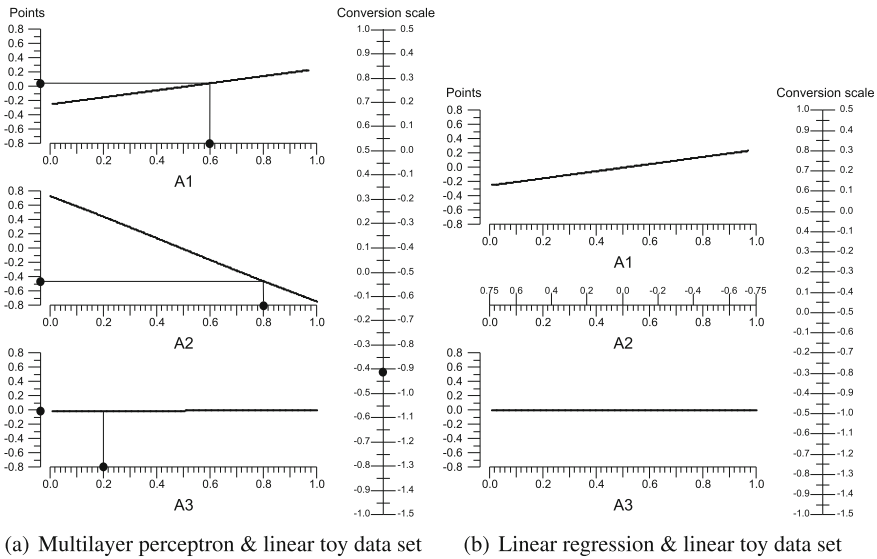


Fig. 8.4 Quasi-nomograms for two different types of models and the same linear problem data set. For this illustrative data set the input variable A_3 is irrelevant to the target variable, which is clear from its flat effect function

model, the transformation g is, at least to a practical precision, the same as the original model ($e_{f,g} = 1.1 \times 10^{-11}$). The quasi-nomogram for the *linear regression* model is shown in Fig. 8.4b and can be used to compare the two different models. Because the structure of the model is not known, A_3 is included in the nomogram, despite being irrelevant. However, the irrelevance of input variable A_3 results in a flat effect function.

Note that input variables with finite domains (see Figs. 8.1 or 8.5b) can be visualised in a more compact way. That is, listing the variable’s values on a single axis, as opposed to a 2-dimensional plot. The same applies to continuous input variables for which a visual inspection reveals a linear (or monotonic) effect function (see Fig. 8.4b, input variable A_2 , or Fig. 8.6, several variables). For such variables, we can reduce the visualisation by projecting the labelled values onto the x axis as it is clear how to interpolate the effect of in-between values.

Quasi-nomograms are useful even when the effect of input variables is not linear. Consider the second toy data set with two input variables A_1 and A_2 with continuous domains $[0, 1]$ and uniformly distributed values. The target variable is defined as $Y = \sin(2\pi A_1) + A_2$. Again, we generated 1000 instances from this data set and used *bagging* to train an ensemble of regression trees. The ensemble gave the following results ($e_{f,y} = 0.048$, $e_{g,y} = 0.041$, $e_{f,g} = 0.034$). For comparison, the results for linear regression were ($e_{f,y} \approx e_{g,y} = 0.43$, $e_{f,g} = 6.2 \times 10^{-16}$). Therefore, the transformed bagging model (see Fig. 8.5a) is more accurate than linear regression, while still easily represented with a quasi-nomogram.

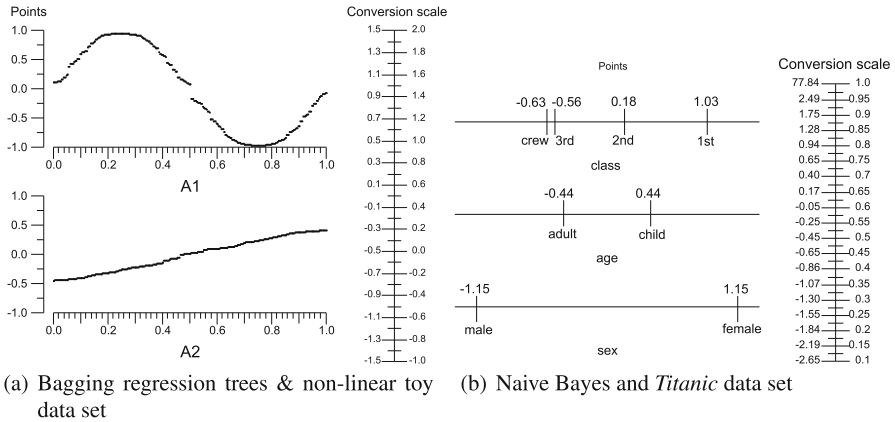


Fig. 8.5 Quasi-nomograms for two different types of models and data sets. The non-linear toy data set has continuous input variables and a continuous target variable. Step effect functions are a characteristic of tree-based models. The *Titanic* data set has discrete input variables and a binomial target variable

Finally, Fig. 8.5b shows the quasi-nomogram for the *Naive Bayes* classifier and the *Titanic* data set. It is equivalent to the quasi-nomogram from the introduction (see Fig. 8.1) that we produced with Orange [7]. That is, the predictions obtained from the two quasi-nomograms for the same instance are equal. For example, if we revisit the adult male travelling first class from Fig. 8.1, but use the quasi-nomogram from Fig. 8.5b instead, we obtain a sum of -0.56 (-1.15 for being male, -0.44 for an adult, and $+1.03$ for travelling first class). Using the conversion scale this sum converts to approximately 45%, which is the same as the prediction obtained from Fig. 8.1. The two nomograms also offer the same insight into the influence of the input variables on survival. For example, being female or travelling first class contributed more towards survival.

8.4.1 Predicting the Outcome of a Basketball Match

For a more realistic illustration of what data-mining practitioners encounter in practice, we performed the following experiment. For each basketball match of the 2007/08 NBA (National Basketball Association) season, we recorded the winner and the following three summary statistics, for both competing teams: effective field goal percentage (EFG), turnover percentage (TOV), and offensive rebounding percentage (ORB). For a more detailed description of these summary statistics see [17].

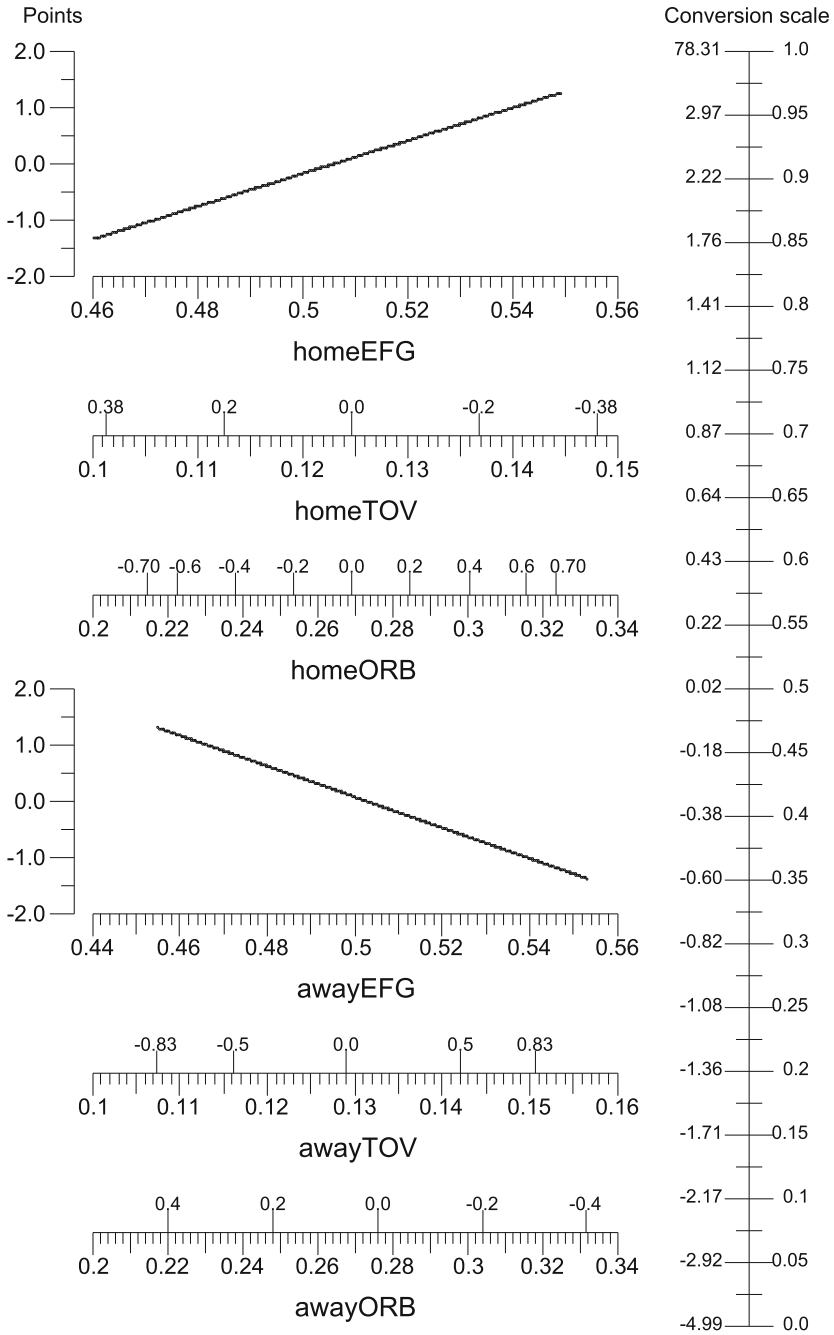


Fig. 8.6 A quasi-nomogram for predicting the win probability of the home team in an NBA basketball match. It is clear from the visualisation that the win probability of the home team increases with its shooting efficiency and decreases with the shooting efficiency of the away team. The remaining four input variables are visualised in a more compact way

This gave us a total of 958 matches with 6 predictor variables and a binomial target variable (match outcome) each. We hypothesised that this data could be used to construct a model which could predict the outcome of NBA basketball matches. Using 10-fold cross-validation, we evaluated several classification models and obtained the following results: *multilayer perceptron* ($e_{f,y} = 0.4554$, percentage of correct predictions = 68.5), *Naive Bayes* ($e_{f,y} = 0.4513$, percentage of correct predictions = 68.7), *bagging* ($e_{f,y} = 0.4565$, percentage of correct predictions = 67.3), and *logistic regression* ($e_{f,y} = 0.443$, percentage of correct predictions = 69.4). Note that the relative frequency of home team win was 0.6, so all four models outperform this default prediction for home team win probability.

Out of all the models, logistic regression gave the best results. Because this model is a generalised additive model (with a log-odds ratio link function $F(x) = \ln(\frac{x}{1-x})$), its transformation g is an accurate representation of the original model ($e_{f,g} = 2.7 \times 10^{-16}$). The resulting quasi-nomogram is shown in Fig. 8.6. It can be used both to predict the winner of future matches and to inspect the effect of individual summary statistics on the outcome of a basketball match.

All the learning algorithms we used in our experiments were from the Weka data mining Java library [12], with the exception of the Naive Bayes that we used for Fig. 8.1.

8.5 Conclusion

In our previous work, which we described in Sect. 8.2, we have proposed a method for explaining an arbitrary prediction model in the form of contributions of individual features. In this chapter we extended that work by showing how such a black-box approach to explanation is closely connected to visualising the model in the form of a nomogram or quasi-nomogram. We proposed a method for constructing a quasi-nomogram for a black-box prediction model, only by changing the inputs and observing the changes of the output. This is convenient in situations when working with and comparing several different types of models, when we wish to avoid implementing a model-specific method for constructing a quasi-nomogram, or when such an implementation is not possible (only the model might be available or we want to avoid investing the time and effort necessary for modifying third-party code).

These quasi-nomograms can also be viewed as prediction curve plots (one for each variable) - plots of how the effect of the variable depends on its value. The only difference is that the values of other variables are not fixed, but varied in a way that captures interactions among all subsets of variables (see Sect. 8.2). If the prediction curve is monotonic (strictly increasing or decreasing) or the variable has only a few values, it can be collapsed onto a single axis, as was shown in some of the examples.

As shown in the illustrative examples, the procedure can be applied to classification and regression tasks, discrete and continuous input variables. For models which are known to be of the generalised additive type, the method is interchangeable with any model-specific method, as it produces equivalent quasi-nomograms,

and therefore generalises them. The approach is useful for non-additive models as well, especially when the task is additive or when the resulting loss in prediction accuracy is small or outweighed by the benefits with respect to the interpretability offered by the quasi-nomogram. Additionally, non-additivity is straightforward to detect from the variability and it would be possible to visualise the most important pairwise interaction, further improving how the nomogram represents the model. Three-way or higher level interactions become problematic, however, due to the difficulties of effectively visualising, or representing in some other way, three (and higher) dimensional data and for the human to understand such representations.

References

1. Achen, C.H.: *Intepreting and Using Regression*. Sage Publications (1982)
2. Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., MÄžller, K.R.: How to explain individual classification decisions. *J. Mach. Learn. Res.* **11**, 1803–1831 (2010)
3. Bosnić, Z., Vračar, P., Radović, M.D., Devedzić, G., Filipović, N.D., Kononenko, I.: Mining data from hemodynamic simulations for generating prediction and explanation models. *IEEE Trans. Inf. Technol. Biomed.* **16**(2), 248–254 (2012)
4. Breiman, L.: Random forests. *Mach. Learn. J.* **45**, 5–32 (2001)
5. Cho, B.H., Yu, H., Lee, J., Chee, Y.J., Kim, I.Y., Kim, S.I.: Nonlinear support vector machine visualization for risk factor analysis using nomograms and localized radial basis function kernels. *IEEE Trans. Inf. Technol. Biomed.* **12**(2), 247–256 (2008)
6. Chun, F.K.H., Briganti, A., Karakiewicz, P.I., Graefen, M.: Should we use nomograms to predict outcome?. *Eur. Urol. Suppl.* **7**(5), 396–399 (2008). Update Uro-Oncology 2008, Fifth Fall Meeting of the European Society of Oncological Urology (ESOU)
7. Demšar, J., Zupan, B., Leban, G., Curk, T.: Orange: From experimental machine learning to interactive data mining. In: *PKDD'04*, pp. 537–539 (2004)
8. d'Ocagne, M.: *Traité de nomographie*. Gauthier-Villars, Paris (1899)
9. Doerfler, R.: The lost art of nomography. *UMAP J.* **30**(4), 457–493 (2009)
10. Eastham, J.A., Scardino, P.T., Kattan, M.W.: Predicting an optimal outcome after radical prostatectomy: the trifecta nomogram. *J. Urol.* **79**(6), 2011–2207 (2008)
11. Grömping, U.: Estimators of relative importance in linear regression based on variance decomposition. *Am. Stat.* **61**(2), (2007)
12. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. *SIGKDD Explor. Newsl.* **11**(1), 10–18 (2009)
13. Jaeckel, P.: *Monte Carlo Methods in Finance*. Wiley, New York (2002)
14. Jakulin, A., Možina, M., Demšar, J., Bratko, I., Zupan, B.: Nomograms for visualizing support vector machines. In: *KDD '05: Proceeding of the eleventh ACM SIGKDD International Conference on Knowledge Discovery In Data Mining*, pp. 108–117. ACM, New York, USA (2005)
15. Kanao, K., Mizuno, R., Kikuchi, E., Miyajima, A., Nakagawa, K., Ohigashi, T., Nakashima, J., Oya, M.: Preoperative prognostic nomogram (probability table) for renal cell carcinoma based on tnm classification. *J. Urol.* **181**(2), 480–485 (2009)
16. Kattan, M.W., Marasco, J.: What is a real nomogram. *Semin. Oncol.* **37**(1), 23–26 (2010)
17. Kubatko, J., Oliver, D., Pelton, K., Rosenbaum, D.T.: A starting point for analyzing basketball statistics. *J. Quantit. Anal. Sports* **3**(3), 00–01 (2007)
18. Kukar, M., Grošelj, C.: Supporting diagnostics of coronary artery disease with neural networks. In: *Adaptive and Natural Computing Algorithms*, pp. 80–89. Springer, Berlin (2011)

19. Kukar, M., Kononenko, I., Grošelj, C.: Modern parameterization and explanation techniques in diagnostic decision support system: a case study in diagnostics of coronary artery disease. *Artif. Intell. Med.* **52**(2), 77–90 (2011)
20. Lee, K.M., Kim, W.J., Ryu, K.H., Lee, S.H.: A nomogram construction method using genetic algorithm and naive Bayesian technique. In: *Proceedings of the 11th WSEAS International Conference on Mathematical and Computational Methods In Science And Engineering*, pp. 145–149. World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA (2009)
21. Lemaire, V., Féraud, R., Voisine, N.: Contact personalization using a score understanding method. In: *International Joint Conference on Neural Networks (IJCNN)* (2008)
22. Lughofer, E., Richter, R., Neissl, U., Heidl, W., Eitzinger, C., Radauer, T.: Advanced linguistic explanations of classifier decisions for users' annotation support. In: *2016 IEEE 8th International Conference on Intelligent Systems (IS)*, pp. 421–432. IEEE, New York (2016)
23. Možina, M., Demšar, J., Kattan, M., Zupan, B.: Nomograms for visualization of naive Bayesian classifier. In: *PKDD '04: Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 337–348. Springer, New York, USA (2004)
24. Nguyen, C.T., Stephenson, A.J., Kattan, M.W.: Are nomograms needed in the management of bladder cancer?. *Urol. Oncol. Semin. Orig. Investig.* **28**(1), 102 – 107 (2010). *Proceedings: Midwinter Meeting of the Society of Urologic Oncology (December 2008): Updated Issues in Kidney, Bladder, Prostate, and Testis Cancer*
25. Niederreiter, H.: Low-discrepancy and low-dispersion sequences. *J. Number Theory* **30**(1), 51–70 (1988)
26. Niederreiter, H.: *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (1992)
27. Pregelj, M., Štrumbelj, E., Mihelcic, M., Kononenko, I.: Learning and explaining the impact of enterprises organizational quality on their economic results. *Intelligent Data Analysis for Real-Life Applications: Theory and Practice* pp. 228–248 (2012)
28. Radović, M.D., Filipović, N.D., Bosnić, Z., Vračar, P., Kononenko, I.: Mining data from hemodynamic simulations for generating prediction and explanation models. In: *2010 10th IEEE International Conference on Information Technology and Applications in Biomedicine (ITAB)*, pp. 1–4. IEEE, New York (2010)
29. Robnik-Šikonja, M., Kononenko, I.: Explaining classifications for individual instances. *IEEE Trans. Knowl. Data Eng.* **20**(5), 589–600 (2008)
30. Robnik-Šikonja, M., Kononenko, I., Štrumbelj, E.: Quality of classification explanations with prbf. *Neurocomputing* **96**, 37–46 (2012)
31. Robnik-Šikonja, M., Likas, A., Constantinopoulos, C., Kononenko, I., Štrumbelj, E.: Efficiently explaining decisions of probabilistic RBF classification networks. In: *Adaptive and Natural Computing Algorithms*, pp. 169–179. Springer, Berlin (2011)
32. Robnik-Šikonja, M., Kononenko, I.: Explaining classifications for individual instances. *IEEE TKDE* **20**, 589–600 (2008)
33. Shapley, L.S.: *A Value for n-person games. Contributions to the Theory of Games*, vol. II. Princeton University Press, Princeton (1953)
34. Štrumbelj, E., Bosnić, Z., Zakotnik, B., Grašič-Kuhar, C., Kononenko, I.: Explanation and reliability of breast cancer recurrence predictions. *Knowl. Inf. Syst.* **24**(2), 305–324 (2010)
35. Štrumbelj, E., Kononenko, I.: An efficient explanation of individual classifications using game theory. *J. Mach. Learn. Res.* **11**, 1–18 (2010)
36. Štrumbelj, E., Kononenko, I.: A general method for visualizing and explaining black-box regression models. In: Dobnikar A., Lotric U., Ster B. (eds.) *ICANNNGA (2). Lecture Notes in Computer Science*, vol. 6594, pp. 21–30. Springer, Berlin (2011)
37. Štrumbelj, E., Kononenko, I.: Explaining prediction models and individual predictions with feature contributions. *Knowl. Inf. Syst.* **41**(3), 647–665 (2014)
38. Vien, N.A., Viet, N.H., Chung, T., Yu, H., Kim, S., Cho, B.H.: Vrifa: a nonlinear SVM visualization tool using nomogram and localized radial basis function (LRBF) kernels. In: *CIKM*, pp. 2081–2082 (2009)

39. Zien, A., Krämer, N., Sonnenburg, S., Rätsch, G.: The feature importance ranking measure. In: ECML PKDD 2009, Part II, pp. 694–709. Springer, Berlin (2009)
40. Zlotnik, A., Abaira, V.: A general-purpose nomogram generator for predictive logistic regression models. *Stata J.* **15**(2), 537–546 (2015)