Explanation and Reliability of Individual Predictions

Igor Kononenko, Erik Štrumbelj, Zoran Bosnić, Darko Pevec, Matjaž Kukar, Marko Robnik-Šikonja
Laboratory for Cognitive Modeling (LKM)
University of Ljubljana, Faculty of Computer and Information Science
Tržaška 25, 1000 Ljubljana, Slovenia
e-mail: lkm@fri.uni-lj.si (name.surname@fri.uni-lj.si); www: lkm.fri.uni-lj.si

Keywords: explanation, reliability, individual predictions, prediction intervals

Received: December 15, 2012

Classification and regression models, either automatically generated from data by machine learning algorithms, or manually encoded with the help of domain experts, are daily used to predict the labels of new instances. Each such individual prediction, in order to be accepted/trusted by users, should be accompanied by an explanation of the prediction as well as by an estimate of its reliability. We have recently developed a general methodology for explaining individual predictions as well as for estimating their reliability. Both, explanation and reliability estimation are general techniques, independent of the underlying model and provide on-line (effective and efficient) support to the users of prediction models.

Povzetek: Razvita je metodologija za razlago napovedi s pripadajočo verjetnostjo pri klasifikacijskih in regresijskih modelov strojnega učenja.

1 Introduction

In a typical machine learning scenario a machine learning algorithm is used to construct a model of the relationships between the input features and the target variable with the purpose to predict the target variable of new, yet unseen instances. Explaining the learned relationships is also an important part of machine learning. Some models, such as additive models or small decision trees, are inherently transparent and require little or no additional post processing [14, 32]. Other, more complex and often better performing models are non-transparent and require additional explanation. Therefore, model-specific explanation methods have been developed for models such as artificial neural networks and SVM. In practice, dealing with several different explanation methods requires undesirable additional effort and makes it difficult to compare models of different types. To address this issue, general explanation methods are used – methods, which treat each model as a black-box and can be used independent of the model's type. Most general explanation methods are based on marginalization of features [17, 35]. This approach is computationally efficient. It is also effective as long as the model is additive (that is, as long as the features do not interact). However, several widely-used machine learning models are not additive, which leads to misleading and incorrect explanations of the importance and influence of features [29]. Unlike existing general explanation methods, our method, described in Section 2, takes into account not only the marginal effect of single features but also the effect of subsets of features.

In supervised learning, one of the goals is to get the best possible prediction accuracy on new and unknown instances. As current prediction systems do not provide enough information about single predictions, experts find it hard to trust them. Common evaluation methods for classification and regression machine learning models give an averaged accuracy assessment of models, and in general, predictive models do not provide reliability estimates for their individual predictions. In many areas, appropriate reliability estimates may provide additional information about the prediction correctness and can enable the user (e.g. medical doctor) to differentiate between more and less reliable predictions. In Section 3 we describe our approaches to estimating the reliability of individual predictions. Finally, in Section 4 we overview directions of current research, carried out in LKM.

2 Explaining individual predictions

The idea behind our method for explaining individual predictions is to compute the contributions of individual features to the model's prediction for a particular instance by decomposing the difference between the model's prediction for the given instance and the model's expected prediction (i.e., the model's predictions if none of the features' values were known). We adopt, with minor modifications, the notation used in [30]. Let $A = A_1 \times A_2 \times \cdots \times A_n$ be our feature space, where each feature A_i is a set of values. Let p be the probability mass function defined on the sample space A. Let $f_c : A \to [0, 1]$ describe the classification model's prediction for class value c. Our goal is a general explanation method which can be used with any model, so no other assumptions are made about f_c. Therefore, we are limited to changing the inputs of the model and observing the outputs.

Let $S = A_1, \dots, A_n$ be the set of all features. The influence of a certain subset Q of S for classification of a given instance is defined as:

$$\Delta(Q)(x) = E\left[f|Q(x)\right] - E\left[f\right] \qquad (1)$$

where $Q(x)$ are values of features in Q for x. The value of the above function for the entire set of features S is exactly the difference between the model's prediction for a given instance and the model's expected prediction that we wish to decompose. Note that we omit the class value in the notation of f. Suppose that for every subset of features Q the value of $\Delta(Q)$ is known. The goal is to decompose $\Delta(S)$ in a way that assigns each feature a fair contribution with respect its influence on the model's prediction. In [29] a solution is proposed that is equivalent to the Shapley value [27] for the coalition game with the n features as players and Δ as the characteristic function. The contribution of the i-th feature is defined as

$$\phi_i(x) = \sum_{Q \subseteq S_{\{i\}}} \frac{|Q|!\,(|Q|-1)!}{|S|!} \delta(Q, x, i), \qquad (2)$$

where

$$\delta(Q, x, i) = \Delta(Q \cup \{i\})(x) - \Delta(Q)(x). \qquad (3)$$

These contributions have some desirable properties. Their sum for the given instance x equals $\Delta(S)$, which was our initial goal and ensures implicit normalization. A feature that does not influence the prediction will be assigned no contribution. And, features that influence the prediction in a symmetrical way will be assigned equal contributions.

The computation of Eq. 2 is infeasible for large n as the computation time grows exponentially with n. The approximation algorithm is proposed in [31, 30], where we show its efficiency and effectiveness. It is based on the assumption that p is such that individual features are mutually independent. With this assumption and using an alternative formulation of the Shapley value we get a formulation which facilitates random sampling. For a global view on features' contributions, we define the contribution of the feature's value as the expected value of that feature's contribution for a given value. Again, random sampling can be used to estimate the expected value [30]. Let us illustrate the use of the features' local and global contributions using a simple data set with 5 numerical features $A_1, \dots, A_5$ with unit domains $[0, 1]$. The binary class value equals 1 if $A_1 > 0.5$ or $A_2 > 0.7$ or $A_3 > 0.5$. Otherwise, the class value is 0. Therefore, only the first three features are relevant for predicting the class value. This problem can be modeled with a decision tree. Figure 1 shows explanations for such a decision tree. The global contributions of each feature's values are plotted separately. The black line consists of points obtained by running the approximation algorithm for the corresponding feature and its value

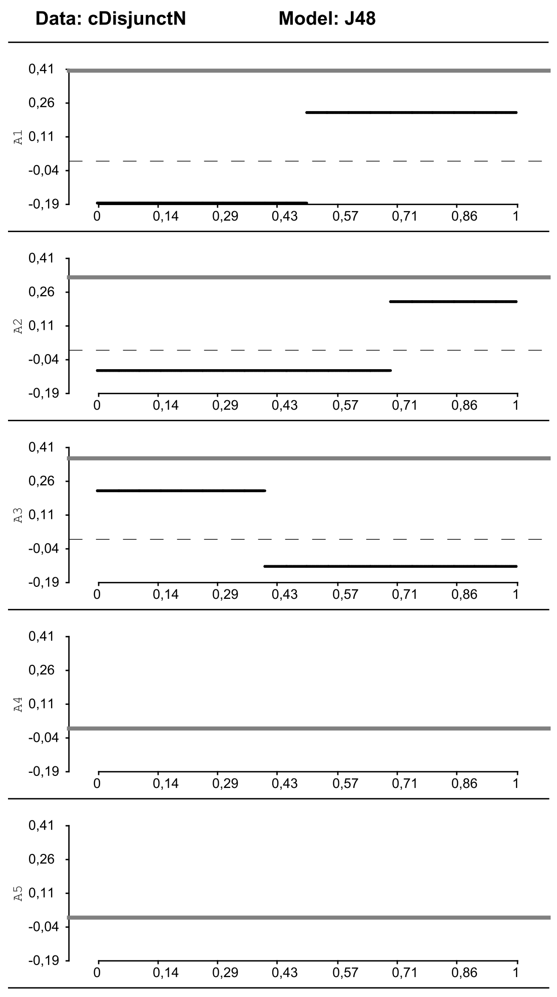

Figure 1: Explanation of a decision tree.

corresponding the value on the x-axis. The lighter line corresponds to the standard deviation of the samples across all values of that particular feature and can therefore be interpreted as the overall importance of the feature. The lighter lines reveal that only the first three features are important. The black lines reveal the areas where features contribute towards/against class value 1. For example, if the value of feature A_1 is higher than 0.5 it strongly contributes towards class value being 1. If it is lower, it contributes against class value being 1. For example, the instance $x = (0.47, 0.82, 0.53, 0.58, 0.59)$ belongs to class 1, which the decision tree correctly predicts. The visualization on Figure 2 shows the individual features' contributions for this instance. The last two features have a 0 contribution. The only feature value that contributes towards class = 1 is $A_2 = 0.82$, while the remaining two features' values have a negative contribution.

We have successfully applied this research to post-processing tools for breast cancer recurrence prediction [31], maximum shear stress prediction from hemodynamic simulations [6], and businesses' economic results precision [24].

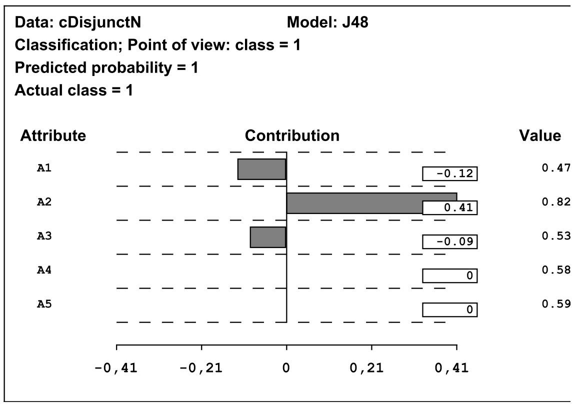

Figure 2: Visualization of the individual features' contributions for particular instance.

2.1 Efficient RBF network explanation

A lot of effort has been invested into increasing the interpretability of complex models such as artificial neural networks [2, 21]. Presented explanations method can be used with any type of classification model including neural networks. Below we show how for a special type of neural networks the approximations required to estimate Eq. 1 in [31, 30] can be especially efficient and require no relearning of the classification model [26]

The probabilistic radial basis function network classifier (PRBF) is an effective but non-transparent prediction model [11, 33]. The PRBF is a special case of the RBF network [3]. It adopts a cluster interpretation of the base functions, where each cluster can generate observations for any class. Therefore, it is a generalization of the Gaussian mixture model [3, 20]. We show how the PRBF can be efficiently explained with two presented explanation methods [25, 29].

Consider a classification problem with c classes y_k $(k = 1, \ldots, c)$ and input instances $x = (A_1, \ldots, A_a)$. For this problem, the corresponding PRBF classifier has a inputs and c outputs, one for each class. Each output provides and estimate of the probability density $p(x|y_k)$ of the corresponding class y_k. Assume that we have M components (hidden units), each one computing a probability density value $f_j(x)$ of the input x. In the PRBF network all component density functions $f_j(x)$ are utilized for estimating the conditional densities of all classes by considering the components as a common pool [33]. Thus, for each class a conditional density function $p(x|y_k)$ is modeled as a mixture model of the form:

$$p(x|y_k) = \sum_{j=1}^{M} \pi_{jk} f_j(x), \quad k = 1, \ldots, c, \tag{4}$$

where the mixing coefficients π_{jk} are probability vectors; they take positive values and satisfy the following constraint:

$$\sum_{j=1}^{M} \pi_{jk} = 1, \quad k = 1, \ldots, c. \tag{5}$$

Once the outputs $p(x|y_k)$ have been computed, the class of data point x is determined using the Bayes rule, i.e. x is assigned to the class with maximum posterior $p(y_k|x)$ computed by

$$p(y_k|x) = \frac{p(x|y_k)P_k}{\sum_{\ell=1}^{c} p(x|y_\ell)P_\ell} \tag{6}$$

The class priors P_k are usually computed as the percentage of training instances belonging to class y_k.

In the following, we assume the Gaussian component densities of the general form:

$$f_j(x) = \frac{1}{(2\pi)^{a/2}|\Sigma_j|^{1/2}} \cdot \exp\left\{-\frac{1}{2}(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j)\right\} \tag{7}$$

where $\mu_j \in \Re^a$ represents the mean of component j, while Σ_j represents the corresponding $a \times a$ covariance matrix. The whole adjustable parameter vector of the model consists of the mixing coefficients π_{jk} and the component parameters (means μ_j and covariances Σ_j).

A notable convenient characteristic of the Gaussian distribution is the marginalization property: if the joint distribution of a set of random variables $S = \{A_1, \ldots, A_a\}$ is Gaussian with mean μ and covariance matrix Σ, then for any subset A of these variables, the joint distribution of the subset $Q = S - A$ of the remaining variables is also a Gaussian. The mean $\mu_{\setminus A}$ of this Gaussian is obtained by removing from μ the components corresponding to the variables in subset A and covariance matrix $\Sigma_{\setminus A}$ is obtained by removing the rows and columns of Σ corresponding to the variables in subset A. Therefore, if we know the mean and covariance of the joint distribution of a set of variables, we can immediately obtain the distribution of any subset of these variables.

The basis for the explanation is $\Delta(Q)(x)$ from Eq. 1, so for a given instance x we need the prediction using the set of all features $\{1, 2, ...a\}$, the prediction using only a subset of features Q, and the prediction using the empty set of features $\{\}$. Let these predictions be $h(x_{\{1,2,...a\}})$, $h(x_Q)$, and $h(x_{\{\}})$, respectively. Since an instance x is fixed in explanation, for readability sake we omit it from expressions below, but remain aware that the dependence exists.

For an input $x = (A_1 = v_1, \ldots, A_a = v_a)$ each output $p(x|y_k)$, $k = 1, \ldots, c$ of the PRBF is computed as a mixture of Gaussians:

$$p(x|y_k) = \sum_{j=1}^{M} \pi_{jk} \mathcal{N}(x; \mu_j, \Sigma_j) \tag{8}$$

Consequently, based on the marginalization property of the Gaussian distribution, for a subset of features Q we get

$$h(x_Q|y_k) = \sum_{j=1}^{M} \pi_{jk}\mathcal{N}(x_Q;\mu_{jQ},\Sigma_{jQ}) \tag{9}$$

where μ_{jQ} and Σ_{jQ} are obtained by retaining only the elements from Q in μ_j and Σ_j. We obtain $h(y_k|x_Q)$ as

$$h(y_k|x_Q) = \frac{h(x_Q|y_k)P_k}{\sum_{\ell=1}^{c} h(x_Q|y_\ell)P_\ell} \tag{10}$$

and use it in the approximation via Eqs. (2) and (2).

As a consequence, with PRBF models approximations become more efficient and exact as no approximation of the conditional predictions is needed. Classification with a subset of features requires only a mask which selects appropriate elements from μ and appropriate rows and columns from Σ matrix.

3 Reliability of individual predictions

Because model independent approaches are general, they cannot exploit specific parameters of a given predictive model, but rather focus on influencing the parameters that are available in the standard supervised learning framework (e.g. a learning set and attributes). We expect from reliability estimators to give insight into the prediction error and we expect to find positive correlation between the two. There exist two distinct types of estimates – point estimators are covered in the first subsection and interval estimators are presented in the latter.

3.1 Point estimators

The first two algorithms described in this section are based on Reverse transduction and Local sensitivity analysis and follow the same transductive idea, though the first is applicable only to classification, and the second to regression models. Other algorithms are general and need only minor adaptations when converted from regression to classification models, or vice versa.

3.1.1 Reverse transduction and Local sensitivity analysis

Transduction can be used in the reverse direction, in the sense of observing the model's behavior when inserting modified learning instances of the unseen and unlabeled instance [15, 16]. Let x represent an instance and let y be its class label. Let us denote a learning instance with known label y as (x, y) and let $(x, \_)$ be an unseen and unlabeled instance, for which we wish to estimate the reliability of an initial prediction K. It is possible to create a modified learning instance by inserting the unseen instance $(x, \_)$ into the learning set and label it with the same (first) or different class (second best or last) as predicted by the initial model. The distance between the initial probability vector and that from the rebuilt model forms the reliability estimate. The three reliability estimators for classification models derived from three instances are labeled $TRANS_{first}$, $TRANS_{second}$ and $TRANS_{last}$.

In the regression the procedure is similar except that the predicted label is first slightly corrupted: $y = K + \delta$ and then we insert the newly generated instance (x, y) into the learning set and rebuild the predictive model. We define $\delta = \epsilon(l_{\max} - l_{\min})$, where ϵ expresses the proportion of the distance between largest ($l_{\max}$) and smallest ($l_{\min}$) prediction. In this way we obtain a sensitivity model, which computes a sensitivity estimate K_ϵ for the instance $(x, \_)$. To widen the observation window in local problem space and make the measures robust to local anomalies, the reliability measures use estimates from the sensitivity models, gained and averaged across different values of $\epsilon \in E$. For more details see [4]. Let us assume we have a set of nonnegative ϵ values $E = \epsilon_1, \epsilon_2, \ldots, \epsilon_{|E|}$. We define the estimates as follows:

– Estimate $SAvar$
(Sensitivity Analysis local variance):

$$SAvar = \frac{\sum_{\epsilon \in E}(K_\epsilon - K_{-\epsilon})}{|E|} \tag{11}$$

– Estimate $SAbias$
(Sensitivity Analysis local bias):

$$SAbias = \frac{\sum_{\epsilon \in E}(K_\epsilon - K) + (K_{-\epsilon} - K)}{2\,|E|} \tag{12}$$

3.1.2 Bagging variance

In related work, the variance of predictions in the bagged aggregate of artificial neural networks has been used to estimate the reliability of the aggregated prediction [13, 9]. The proposed reliability estimate is generalized to other models [5].

Let $K_i, i = 1 \ldots m$, be the predictor's class probability distribution for a given unlabeled example $(x, \_)$. Given a bagged aggregate of m predictive models, where each of the models yields a prediction $B_k, k = 1 \ldots m$, the reliability estimator $BAGV$ is defined as the variance of predictions' class probability distribution:

$$BAGV = \frac{1}{m}\sum_{k=1}^{m}\sum_{i}(B_{k,i} - K_i)^2\,. \tag{13}$$

The algorithm uses a bagged aggregate of 50 predictive models as default.

3.1.3 Local cross-validation

The LCV (Local Cross-Validation) reliability estimate is computed using the local leave-one-out (LOO) procedure. Focusing on the subspace defined by k nearest neighbors, we generate k local models, each of them excluding one of the k nearest neighbors. Using the generated models, we compute the leave-one-out predictions $K_i, i = 1 \dots k$, for each of the k excluded nearest neighbors. Since the labels $C_i, i = 1 \dots k$, of the nearest neighbors are known, we are able to calculate the local leave-one-out prediction error as the average of the nearest neighbors' local errors:

$$LCV = \frac{1}{k} \sum_i |C_i - K_i| \,. \qquad (14)$$

In experiment, the parameter k was assigned to one tenth of the size of the learning set.

3.1.4 Local error modeling

Given a set of k nearest neighbors, where C_i is the true label of the i-th nearest neighbor, the estimate CNK ($C_{\mathrm{N}eighbors} - K$) is defined as the difference between average label of the k nearest neighbors and the instance's prediction K:

$$CNK = \frac{\sum_i C_i}{k} - K. \qquad (15)$$

CNK is not a suitable reliability estimate for the k-nearest neighbors algorithm, as they both work by the same principle. In our experiments we used $k = 5$. In regression tests, *CNK-a* denotes the absolute value of the estimate, whereas *CNK-s* denotes the signed value.

3.1.5 Density based estimation

This approach assumes that an error is lower for predictions in denser problem subspaces, and higher for predictions in sparser subspaces. Note that it does not consider the learning instances' labels. The reliability estimator DENS is a value of the estimated probability density function for a given unlabeled example.

3.1.6 Applications of point estimators

The proposed reliability estimation methodology has been implemented in several applications of machine learning and data mining in areas of medicine, financial applications, economy. In these application domains, the bare regression predictions have been supplemented with a suitable reliability estimator (the best performing among the proposed was chosen after the initial evaluation study), which helped the users of predictive systems gain greater insight into the trustworthiness of individual predictions. The most interesting of these applications are:

– breast cancer recurrence prediction problem for the Institute of Oncology, Ljubljana [31]. The collected dataset included data for 1023 patients, for whom the task was to predict potential cancer recurrence for the period of future 20 years. Given that a predictive timespan is so wide and that it reaches into the far future, the difficulty of the predictive problem was alleviated by implementing reliability estimators,
– electricity load forecast prediction problem for a particular European country [6]. Two regression models were implemented, the neural network and the k nearest neighbors algorithm and their predictions were corrected using the reliability estimator CNK. The results showed that the accuracy of corrected predictions using CNK is favorable in comparison to accuracy of predictions corrected with referential Kalman filter method.
– predicting maximum wall shear stress magnitude and its coordinates in the model of human carotid artery bifurcation [7]. Since one of the most common causes of human death is stroke, a medical expert system could significantly aid medical experts to detect hemodynamic abnormalities. Based on the acquired simulated data, we applied several prediction reliability estimators and the model explanation methodology that provided a useful tool for the given problem domain.

3.2 Interval estimators

Here we focus on standard regression problems where the data follows some continuous function and is somehow corrupted with additive noise – $y_i = f(\vec{x_i}) + \epsilon(\vec{x_i})$.

Confidence intervals are concerned with the accuracy of the model's estimate $\hat{y}_i = \hat{y}(\vec{x_i})$ of the true but unknown function $f(\vec{x_i})$. They strive to capture the distribution of the quantity $f(\vec{x_i}) - \hat{y}_i$, however in applications, it is more informative to quantify the accuracy of the model's output with respect to the realized observations y_i. Prediction intervals (PIs) should capture the distribution of individual future points and are concerned with the quantity $y_i - \hat{y}_i$. Expanding the first term, $y_i - \hat{y}_i = f(\vec{x_i}) + \varepsilon(\vec{x}) - \hat{y}_i = [f(\vec{x_i}) - \hat{y}_i] + \varepsilon(\vec{x})$, we see that the PI should enclose the confidence interval. In real world applications, PIs are more practical than confidence intervals because the former are concerned with the accuracy with which it is possible to predict the observed value itself, and not only with the accuracy of the estimate of the true conditional mean.

3.2.1 Bootstrap and maximum likelihood

The first family of methods is based on the idea of explaining the total prediction error as a sum of the model's error and the error caused by noise inherent to the data [13]. Noise in data and non-uniform distribution of examples represent a challenge for learning algorithms, leading to different prediction accuracies in different parts of the problem space. This component is called the data noise variance and is labeled as σ_d^2. Apart from the distribution of learning examples there are also other causes that influence the accuracy of prediction models and these factors form the component called the model uncertainty variance, σ_m^2. The two components are assumed to be independent of each other and their sum is the total prediction variance: $\sigma^2 = \sigma_m^2 + \sigma_d^2$.

The most straightforward approach was formed in [34]. Given a fixed model with available learning algorithm and training set, the reinterpretation of the method goes as follows. To estimate $\sigma_m^2(\vec{x})$, bagging is done with the training data, using the model at hand. Confidence intervals are formed by assuming the normal distribution and calculating $\hat{\sigma}_m^2(\vec{x})$, the variance of the bagged predictions. Then a radial basis function network (RBFN) is trained on the residuals of the bagging predictions on the training dataset and is used to provide the estimate $\hat{\sigma}_d^2(\vec{x})$. Assuming normal distribution, $\hat{\sigma}^2(\vec{x})$ equals to $\hat{\sigma}_m^2(\vec{x})+\hat{\sigma}_d^2(\vec{x})$, so the PI is $\hat{y}_{\text{bag}}(\vec{x}) \pm z_{\alpha/2}\hat{\sigma}(\vec{x})$ where $\hat{y}_{\text{bag}}$ is the bagged prediction of the model. This method is labeled as *BagMLa*.

Generalizing the method from [13], we label it *BagMLb*. Here, $\hat{\sigma}_m^2(\vec{x})$ is again obtained by calculating the variance of the bagged model. Estimation of $\sigma_d^2(\vec{x})$ is done by a RBFN trained on the out–of–sample bagged residuals. Assuming a locally normal distribution, the *BagMLb* PI is $\hat{y}(\vec{x}) \pm z_{\alpha/2}\hat{\sigma}(\vec{x})$ where $\hat{y}$ is the model's prediction. Keen readers would notice that *BagMLa* PIs are centered on $\hat{y}_{bag}(\vec{x})$ but with *BagMLb*, the PIs are centered on $\hat{y}(\vec{x})$. The idea is that $\hat{y}_{\text{bag}}(\vec{x})$ provides a more stable estimate of the true function $f(\vec{x})$ than $\hat{y}(\vec{x})$ does.

3.2.2 Local neighborhood

The second family assumes that samples, which are close in the attribute domain, will behave similarly. These approaches estimate the conditional prediction variance with use of the local neighborhoods for direct estimation of σ^2. As such, they can be applied even in cases where there is no access to the learning algorithm or the bootstrap procedure would be just too time consuming.

Adopting the idea from [28], we implemented k-means clustering on the training data. The number of clusters is defined with the common heuristic $k = \sqrt{n/2}$, where n is the size of the training set. *LNcl* PIs are constructed for each cluster directly from its empiric distribution of the residuals, by taking the appropriate percentiles, i.e. the 2.5 and 97.5 percentiles for 95% PIs. For an unseen example, the PI is defined by that of the nearest cluster.

The nearest neighbor algorithm can be used to construct PIs in the following way. First, signed residuals are obtained from the training set. From the nearest neighbors residuals, their mean $\bar{r}$ serves for bias correction and their standard deviation gives us $\hat{\sigma}^2(\vec{x})$. The number of used neighbors is relative to the size of the data set. With method *LN5*, the size of the neighborhood is 5% of the total population. Our second variant *LN100* is computationally even simpler, as it covers the whole (100%) population and is therefore equivalent to analytic methods that assume constant variance. Here we assume the Student's *t*-distribution with degrees of freedom equal to the number of neighbors, so the PIs take the form $\hat{y} + \bar{r} \pm t_{\alpha/2} \cdot \hat{\sigma}^2(\vec{x})$.

Natural progression would suggest use of adaptive neighborhood procedures. Random forests [8] were reinterpreted as a weighted mean of the observed response variables in [18] and generalized to Quantile Regression Forests in [19]. In this case, trees in these ensembles are not pruned and the values of all observations are kept in the leafs. This preserves information about the underlying distribution and makes estimation of conditional quantiles possible. The conditional distribution of residuals given $X = x$ can be written as $F(r|X = x) = P(R \leq r|X = x) = E(1_{R\leq r}|X = x)$, where $1_{R_i\leq r}$ is an indicator variable with value 1 if $R_i \leq r$ and 0 otherwise. This expression is approximated by the weighted mean over the indicator variables and the estimator is

$$\hat{F}(r|X = x) = \sum_{i=1}^{n} w_i(x) 1_{R_i\leq r}.$$

The weights w_i are averaged weights from the collection of trees and a weight in the individual tree is the inverse of the number of observations in the corresponding leaf. Weights sum to one, so they represent the distribution of possible values for the response variable. When a new sample is dropped trough the collection of trees, the weights are obtained. The corresponding r_i values are sorted in ascending order and for 95% PIs, we need to find r_l for which $\sum_{i=1}^{l} w_i \geq 0.025$ and r_u for which $\sum_{i=1}^{u} w_i \geq 0.975$ hold. The interval $[\hat{y} + r_l, \hat{y} + r_u]$ is our sought PI.

3.2.3 Evaluation of interval estimators

In Figure 3 we can see the results on a collection of 36 real-world and artificial datasets. It shows how the lowest achieved PICP values and those closest to the target PICP value are distributed among the used methods. In this regard, *QRF* seems best.

If two methods achieve the same PICP, the one with a lower RMPI value would have more narrow intervals and should be regarded as the better option. Small RMPI values are mostly achieved by *LN* methods due to their nature of producing optimal intervals and extremely low RMPI values are usually accompanied with zero prediction coverage. Largest RMPI values were produced by *BagML* methods, though the corresponding PICP values are 1.0. This means that all test examples enclosed on the account of very wide PIs. According to the figure, *QRF* did not produce any extreme RMPI values and is even in this respect the best method to use.

4 Current research directions

Our current research is focused on several topics. One research area is related to evaluation of ordinal features in the context of surveys and customer satisfaction in marketing, learning of imbalanced classification problems, and applying evolutionary computation to data mining (focused on using ant colony optimization for rule learning). Somewhat different area is spatial data mining of multi-level directed graphs with applications in oceanography [22].

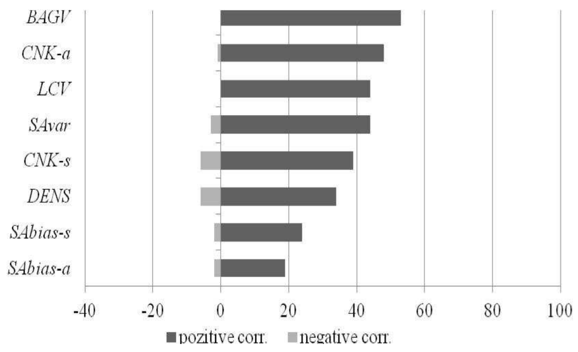

Figure 3: Distribution of experiments (their percentage) of achieved PICP values closest to the target (positive) and furthest (negative) compared to the distribution of the smallest (positive) and largest (negative) achieved RMPI values among the methods.

We recently restored our research in inductive logic programming (ILP) which is focused on employing background knowledge analysis for search space reduction in bottom-up ILP. Yet another branch of research is profiling of web users in an online advertising network together with heuristic search methods in clickstream mining [23], employing algebraic methods, particularly matrix factorization for text summarization, and modeling the progression of team sports matches and evaluation of the individual player's contributions. We also continue our long-term research in medical problems, particularly detection of (non)-ischaemic episodes in ECG signals [12]. The research, described in this paper, continues by adapting the reliability estimators and the explanation methodology for online learning (data streams).

Acknowledgement

Our research is supported by ARRS (Slovenian Research Agency). We thank all former and current members of Laboratory for Cognitive Modeling (LKM) at Faculty of Computer and Information Science, University of Ljubljana, for their contributions.

References

[1] A. Asunction, D. J. Newman (2007) *UCI ML repository*.

[2] A. S. d'Avila Garcez, K. Broda, D. M. Gabbay (2001) Symbolic knowledge extraction from trained neural networks: a sound approach, *Artificial Intelligence 125(1-2)*, pp. 155–207.

[3] C. M. Bishop (1995) *Neural Networks for Pattern Recognition*, Oxford University Press.

[4] Z. Bosnić, I. Kononenko (2007) Estimation of individual prediction reliability using the local sensitivity analysis, *Applied Intelligence, 29(3)*, pp. 187–203.

[5] Z. Bosnić, I. Kononenko (2008) Comparison of approaches for estimating reliability of individual regression predictions, *Data & Knowledge Engineering, 67(3)*, pp. 504–516.

[6] Z. Bosnić, P.P. Rodrigues, I. Kononenko, J. Gama (2011) Correcting streaming predictions of an electricity load forecast system using a prediction reliability estimate, *Man-machine interactions 2*, Springer, pp. 343–350.

[7] Z. Bosnić, P. Vračar, M.D. Radović, G. Devedžić, N. Filipović, I. Kononenko (2012) Mining data from hemodynamic simulations for generating prediction and explanation models, *IEEE trans. inf. technol. biomed., vol. 16, no. 2*, pp. 248–254.

[8] L. Breiman (2001) Random Forests, *Machine Learning – Volume 45*, pp. 5–32.

[9] J. Carney, P. Cunningham (1999) Confidence and prediction intervals for neural network ensembles, *Proceedings of IJCNN'99, The International Joint Conference on Neural Networks*, Washington, USA, pp. 1215–1218.

[10] Department of Statistics at Carnegie Mellon University (2005) *Statlib – Data, software and news from the statistics community* Retrieved from http://lib.stat.cmu.edu/

[11] C. Constantinopoulos, A. Likas (2006) An incremental training method for the probabilistic RBF network, *IEEE Trans. Neural Networks 17(4)*, pp. 966–974.

[12] J. Faganeli Pucer, J. Demšar, M. Kukar (2012) Classification of Ischaemic Episodes with ST/HR Diagrams, *Quality of Life through Quality of Information: Proceedings of MIE2012*, IOS Press, pp. 1108–1111.

[13] T. Heskes (1997) Practical confidence and prediction intervals, *Advances in Neural Information Processing Systems, 9*, The MIT Press, pp. 176–182.

[14] A. Jakulin, M. Možina, J. Demšar, I. Bratko, B. Zupan (2005) Nomograms for visualizing support vector machines, *KDD '05: ACM SIGKDD*, pp. 108–117.

[15] M. Kukar, I. Kononenko (2002) Reliable classifications with machine learning, *Proceedings of Machine Learning: ECML-2002*, Helsinki, Finland, Springer, pp. 219–231.

[16] M. Kukar (2006) Quality assessment of individual classifications in machine learning and data mining, *Knowledge and information systems, vol. 9, no. 3*, pp. 364–384.

[17] V. Lemaire, V. Feraud, N. Voisine (2008) Contact personalization using a score understanding method, *International Joint Conference on Neural Networks (IJCNN)*, pp. 649–654.

[18] Y. Lin, Y. Jeon (2006) Random Forests and Adaptive Nearest Neighbors, *Journal of the American Statistical Association – Volume 101*, pp. 578–590.

[19] N. Meinshausen (2006) Quantile Regression Forests, *Journal of Machine Learning Research – Volume 7*, pp. 983–999.

[20] G. McLachlan and D. Peel (2000) *Finite Mixture Models*, John Wiley & Sons.

[21] V. Palade, C.-D. Neagu, and R. J. Patton (2001) Interpretation of trained neural networks by rule extraction, *Proceedings of the International Conference, 7th Fuzzy Days on Computational Intelligence, Theory and Applications*, London, UK, Springer-Verlag, pp. 152–161.

[22] B. Petelin, V. Malačič, A. Malej, M. Kukar, I. Kononenko (2012) Multi-level association rules and directed graphs for the Lagrangian analysis of the Mediterranean ocean forecasting system (MFS), *Geophys. res. abstr., vol. 14*, pp. 4877.

[23] M. Poženel, V. Mahnič, M. Kukar (2011) Separation of interleaved Web sessions with heuristic search, *Proceedings of ICDM 2010*, IEEE Computer Society, pp. 411–420.

[24] M. Pregeljc, E. Štrumbelj, M. Mihelčič, I. Kononenko (2012) 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.

[25] M. Robnik Šikonja, I. Kononenko (2008). Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering*, 20(5):589-600.

[26] M. Robnik Šikonja, I. Kononenko, E. Štrumbelj (2012). Quality of classification explanations with PRBF. *Neurocomputing* 96:37–46

[27] L.S. Shapley (1953) A Value for n-person Games, *volume II of Contributions to the Theory of Games*, Princeton University Press, pp. 307–317.

[28] D. L. Shrestha, D. P. Solomatine (2006) Machine learning approaches for estimation of prediction interval for the model output, *Neural Networks 19(2)*, pp. 225–235.

[29] E. Štrumbelj, I. Kononenko (2010) An Efficient Explanation of Individual Classifications using Game Theory, *Journal of Machine Learning Research 11*, pp. 1–18.

[30] E. Štrumbelj, I. Kononenko (2011) A General Method for Visualizing and Explaining Black-Box Regression Models, *International Conference on Adaptive and Natural Computing Algorithms (ICANNGA) 2011*, pp. 21–30.

[31] E. Štrumbelj, Z. Bosnić, I. Kononenko, B. Zakotnik, C. Grašič (2010) Explanation and reliability of prediction models: the case of breast cancer recurrence, *Knowledge and information systems, vol. 24, no. 2*, pp. 305–324.

[32] B. Poulin, R. Eisner, D. Szafron, P. Lu, R. Greiner, D.S. Wishart, A. Fyshe, B. Pearcy, C. MacDonell, J. Anvik (2006) Visual explanation of evidence in additive classifiers, *Proceedings of Innovative Applications of Artificial Intelligence – Volume 2*, AAAI Press, pp. 1822–1829.

[33] M. K. Titsias and A. Likas (2001) Shared kernel models for class conditional density estimation, *IEEE Trans. Neural Networks 12(5)*, pp. 987–997.

[34] A. Zapranis, E. Livanis (2005) Prediction intervals for neural network models, *Proceedings of the 9th WSEAS International Conference on Computers*, pp. 76:1–76:7.

[35] A. Zien, N. Krämer, S. Sonnenburg, G. Rätsch (2009) The feature importance ranking measure, *ECML PKDD 2009, Part II*, Springer-Verlag, pp. 694–709.