Explanation of Prediction Models with ExplainPrediction

Marko Robnik-Šikonja
University of Ljubljana, Faculty of Computer and Information Science,
Večna pot 113, 1000 Ljubljana, Slovenia
Email: marko.robnik@fri.uni-lj.si, https://fri.uni-lj.si/en/employees/marko-robnik-sikonja

Keywords: machine learning, comprehensibility of models, explanation of models, perturbation methods

Received: October 31, 2017

State-of-the-art prediction models are getting increasingly complex and incomprehensible for humans. This is problematic for many application areas, especially those where knowledge discovery is just as important as predictive performance, for example medicine or business consulting. As machine learning and artificial intelligence are playing an increasingly large role in the society through data based decision making, this is problematic also from broader perspective and worries general public as well as legislators. As a possible solution, several explanation methods have been recently proposed, which can explain predictions of otherwise opaque models. These methods can be divided into two main approaches: gradient based approaches limited to neural networks, and more general perturbation based approaches, which can be used with arbitrary prediction models. We present an overview of perturbation based approaches, and focus on a recently introduced implementation of two successful methods developed in Slovenia, EXPLAIN and IME. We first describe their working principles and visualizations of explanations, followed by the implementation in ExplainPrediction package for R environment.

Povzetek: Najboljši napovedni modeli postajajo vse bolj zapleteni in nerazumljivi za ljudi. To je problematično za številna aplikativna področja, zlasti tista, kjer je odkrivanje znanja enako pomembno kot napovedna točnost, npr. medicina ali poslovno svetovanje. Ker strojno učenje in umetna inteligenca preko na podatkih temelječega odločanja igrata vse večjo vlogo v družbi, je to problematično tudi s širšega vidika in vse bolj skrbi javnost in zakonodajalce. Kot možna rešitev se je v zadnjem času pojavilo več metod razlage za napovedne modele. Te metode lahko razdelimo na dve skupini: na gradientne metode, omejene predvsem na umetne nevronske mreže, in splošnejše pristope na osnovi perturbacij vhodov, ki jih je mogoče uporabiti pri poljubnih napovednih modelih. Predstavljamo pregled perturbacijskih pristopov in dve uspešni metodi razviti v Sloveniji, EXPLAIN in IME. Najprej opišemo njuno delovanje in vizualizacije razlag, nato pa še implementacijo v paketu ExplainPrediction za okolje R.

1 Introduction

Machine learning methods and especially prediction models are becoming an essential ingredient of many modern products and services. Through a paradigm of data-based decisions, they impact mundane everyday tasks (e.g., shopping and entertainment recommendations), as well as life-changing decisions (e.g., medical diagnostics, credit scoring, or security systems). As societies are getting more and more complex, we can expect that their reliance on automatic decisions will increase. It is natural that those affected by various decisions of prediction models want to get feedback and understand the reasoning process and biases of the underlying models. The impact and influence of automatic decisions are getting so ubiquitous that the whole area of artificial intelligence is receiving an increasing attention from lawmakers who demand that decisions of important models are made transparent. Besides public services, the areas where models' transparency is of crucial importance are for example medicine, science, policy making, strategic planning, business intelligence, finance, marketing, law, and insurance. In these areas, users of models are just as interested in comprehending the decision process, as in the classification accuracy of prediction models. Unfortunately, most of the top performing machine learning models are black boxes in a sense that they do not offer an intrinsic introspection into their decision processes or provide explanations of their predictions and biases. This is true for Artificial Neural Networks (ANN), Support Vector Machines (SVM), and all ensemble methods (for example, boosting, random forests, bagging, stacking, and multivariate adaptive regression splines). Approaches that do offer an intrinsic introspection such as decision trees or decision rules do not perform so well or are not applicable in many cases (17).

To alleviate this problem two types of model explanation techniques have been proposed. The first type, which is not discussed in this work, is based on the internal working of the particular learning algorithm. The explanation methods exploit model's representation or learning process to gain insight into the presumptions, biases, and reasoning leading to final decisions. Two well-known models where

such approach works well are neural networks and random forests. Recent explanations for neural networks classifiers of images mostly rely on layer-wise relevance propagation (3) or gradients of output neurons with respect to the input (28) to visualize parts of images significant for particular prediction. The random forest visualizations mostly exploit the fact that during bootstrap sampling, which is part of this learning algorithm, some of the instances are not selected for learning and can serve as an internal validation set. With the help of this set important features can be identified and similarity between objects can be measured.

The second type of explanation approaches are general and can be applied to any predictive model. The explanations provided by these approaches try to efficiently capture the causal relationship between inputs and outputs of the given model. To this end, they perturb the inputs in the neighborhood of given instance to observe effects of perturbations on model's output. Changes in the outputs are attributed to perturbed inputs and used to estimate their importance for a particular instance. Examples of this approach are methods EXPLAIN (24), IME (31), and LIME (21). These methods can explain models' decision process for each individual predicted instance as well as for the model as a whole. We implemented the methods, proposed by our group, EXPLAIN and IME, in R package ExplainPrediction (23).

The objectives of the paper are twofold. First, we explain how general perturbation-based explanation methods work and second, we describe the implementation details, parameters, and visualization of the ExplainPrediction package which implements them. The first aim is achieved through an explanation of their working principle and graphical explanation of models' decisions on a well-known data set. The second aim is no less important. In machine learning, open source implementations enable progress, empirical comparisons, and replicability of research. Two types of explanations are implemented in ExplainPrediction and demonstrated in the paper, individual predictions of new unlabeled cases and functioning of the model as a whole. This allows inspection, comparison, and visualization of otherwise opaque models.

The structure of the remainder of the paper is as follows. In Section 2, we present the background and related work on perturbation based explanation approaches. In Section 3, we present methods EXPLAIN and IME. Their implementation, parameters, and use with the ExplainPrediction package are covered in Section 4. In Section 5, we present conclusions and promising research directions.

2 Background and overview of perturbation approaches

We first present different modes of explanation and properties of model explanation approaches, followed by an overview of explanation approaches.

True causal relationships between dependent and independent variables are typically hidden except in artificial domains where all the relations, as well as the probability distributions, are known in advance. Therefore only explanations of prediction process for a particular model is of practical importance. The prediction accuracy and the correctness of explanation for a given model may be orthogonal: the correctness of the explanation is independent of the correctness of the prediction. However, empirical observations show that better models (with higher prediction accuracy) enable better explanations (31). We discuss two types of explanations:

- **Instance explanation** explains predictions with the given model of a single instance and provides the impact of input feature values on the prediction.
- **Model explanation** is usually an aggregation of instance explanations over many (training) instances, to provide top-level explanations of features and their values. This aggregation over many instances enables identification of different roles attributes may play in the classifications of instances.

In a typical data science problem setting, users are concerned with both prediction accuracy and the interpretability of the prediction model. Complex models have potentially higher accuracy but are more difficult to interpret. This can be alleviated either by sacrificing some prediction accuracy for a more transparent model or by using an explanation method that improves the interpretability of the model. Explaining predictions is straightforward for symbolic models such as decision trees, decision rules, and inductive logic programming, where the models give an overall transparent knowledge in a symbolic form. Therefore, to obtain the explanations of predictions, one simply has to read the rules in the corresponding model. Whether such an explanation is comprehensive in the case of large trees and rule sets is questionable. Piltaver et al. (18) developed criteria for comprehensibility of decision trees and performed a user study, which showed that the depth of the deepest leaf that is required when answering a question about a classification tree is the most important factor influencing the comprehensibility.

For non-symbolic models, there are no intrinsic explanations. A lot of efforts have been invested in increasing the interpretability of complex models. For SVM, Hamel (12) proposed an approach based on self-organizing maps that groups instances then projects the groups onto a two-dimensional plane. In this plane, the topology of the groups is hopefully preserved and support vectors can be visualized. Many approaches exploit the essential property of additive classifiers to provide more comprehensible explanations and visualizations, e.g., (14) and (19).

Visualization of decision boundaries is an important aspect of model transparency. Barbosa et al. (6) present a technique to visualize how the kernel embeds data into a high-dimensional feature space. With their Kelp method, they visualize how kernel choice affects neighborhood

structure and SVM decision boundaries. Schulz et al. (27) propose a general framework for visualization of classifiers via dimensionality reduction. Goldstein et al. (11) propose another useful visualization tool for classifiers that can produce individual conditional expectation plots, graphing the functional relationship between the predicted response and the feature for individual instance.

Some explanations methods (including the ones presented in Section 3) are general in a sense that they can be used with any type of classification model (15; 21; 24; 30). This enables their application with almost any prediction model and allows users to analyze and compare outputs of different analytical techniques. Lemaire et al. (15) applied their method to a customer relationship management system in the telecommunications industry. The method which successfully deals with high-dimensional text data is presented in (16). Its idea is based on general explanation methods EXPLAIN and IME and offers an explanation in the form of a set of words which would change the predicted class of a given document. Bosnić et al. (9) adapt the general explanation methodology to data stream scenario and show the evolution of attribute contributions through time. This is used to explain the concept drift in their incremental model. In a real-life breast cancer recurrence prediction, Štrumbelj et al. (29) illustrate the usefulness of the visualizations and the advantage of using the general explanation method. Several machine learning algorithms were evaluated. Predictions were enhanced with instance explanations using the IME method. Visual inspection and evaluation showed that oncologists found the explanations useful and agreed with the computed contributions of features. Pregeljc et al. (20) used traditional modeling approaches together with data mining to gain insight into the connections between the quality of organization in enterprises and the enterprises' performance. The best performing models were complex and difficult to interpret, especially for non-technical users. Methods EXPLAIN and IME explained the influence of input features on the predicted economic results and provided insights with a meaningful economic interpretation. The interesting economic relationships and successful predictions come mostly from complex models such as random forests and ANN. Without proper explanation and visualization, these models are often neglected in favor of weaker, but more transparent models. Experts from the economic-organizational field, which reviewed and interpreted the results of the study, agreed that such an explanation and visualization is useful and facilitates comparative analysis across different types of prediction models. Bohanec et al. (7) present an innovative application of explanation methods EXPLAIN and IME in the context of B2B sales forecasting. They demonstrate how users can validate their assumptions with the presented explanations and test their hypotheses using the explanations for a sort of what-if analysis. Bohanec et al. (8) address the problem of weak acceptance of machine learning models in business environments. The propose a framework of top-performing machine learning models coupled with general explanation methods to provide an additional information to the decision-making process. This is shown to reduce error, efficiently support business decision makers and builds a foundation for sustainable organizational learning. Demšar and Bosnić (10) use the general explanation methods EXPLAIN and IME to detect concept drift in data streams. Due to the generality of explanations, their drift detector can be combined with an arbitrary classification algorithm and features good drift detection, accuracy, robustness, and sensitivity.

Many explanation methods are related to statistical sensitivity analysis and uncertainty analysis (26). In that methodology sensitivity of models is analyzed with respect to models' input. A related approach, called inverse classification (1) tries to determine the minimum required change to a data point in order to reclassify it as a member of a different class. An SVM model-based approach is proposed by Barbella et al. (5). Another sensitivity analysis-based approach explains contributions of individual features to a particular classification by observing (partial) derivatives of the classifiers prediction function at the point of interest (4). A limitation of this approach is that the classification function has to be first-order differentiable. For classifiers not satisfying this criterion (for example, decision trees) the original classifier is first fitted with a Parzen window-based classifier that mimics the original one and then the explanation method is applied to this fitted classifier. The method is practically useful with kernel-based classification method to predict molecular features (13).

Due to recent successes of deep neural networks in image recognition and natural language processing, several explanation methods specific to these two application areas emerged, recently. Methods working on images try to visualize parts of images (i.e., groups of pixels) significant for a particular prediction. These methods mostly rely on the propagation of relevance within the network. For example, layer-wise relevance propagation (3), and computation of gradients of output neurons with respect to the input (28). In language processing Arras et al. (2) applied layer-wise relevance propagation to a convolutional neural network and a bag-of-words SVM classifier trained on a topic categorization task. The explanations indicate how much individual words contribute to the overall classification decision.

3 Methods EXPLAIN and IME

General explanation methods can be applied to any classification model which makes them a useful tool both for interpreting models (and their predictions) and comparing different types of models. By modification of feature values of interest, what-if analysis is also supported. Such methods cannot exploit any model-specific properties (e.g., gradients in ANN) and are limited to perturbing the inputs of the model and observing changes in the model's output (15; 24; 30).

The presented general explanation methods provide two types of explanations for prediction models: instance explanations and model explanations (see Section 2). Model explanations work by summarizing a representative sample of instance explanations. The presented methods estimate the impact of particular explanation feature for a given instance by perturbing similar instances.

The key idea of EXPLAIN and IME is that the contribution of a particular input value (or set of values) can be captured by "hiding" the input value (set of values) and observing how the output of the model changes. As such, the key component of general explanation methods is the expected conditional prediction – the prediction where only a subset of the input variables is known. Let Q be a subset of the set of input variables $Q \subseteq S = \{X_1, \ldots, X_a\}$. Let $p_Q(y_k|x)$ be the expected prediction for x, conditional to knowing only the input variables represented in Q:

$$p_Q(y_k|x) = \mathbb{E}(p(y_k)|X_i = x_{(i)}, \forall X_i \in Q). \quad (1)$$

Therefore, $p_S(y_k|x) = p(y_k|x)$. In practical settings, the classification function of the model is not known - one can only access its prediction for any vector of input values. Therefore, exact computation of this equation is not possible and sampling-based approximations are used.

To produce model explanations we sum instance level explanations. The evidence for and against each class is collected and visualized separately. In this way, one can, for example, see that a particular value of an attribute supports specific class but not in every context.

3.1 EXPLAIN, one-variable-at-a-time approach

EXPLAIN method computes the influence of a feature value by observing its impact on the model's output. The EXPLAIN assumes that the larger the changes in the output, the more important role the feature value plays in the model. The shortcoming of this approach is that it takes into account only a single feature at a time, therefore it cannot detect certain higher order dependencies (in particular disjunctions) and redundancies in the model. The EXPLAIN assumes that the characterization of the i-th input variable's importance for the prediction of the instance x is the difference between the model's prediction for that instance and the model's prediction if the value of the i-th variable was not known. The source of explanations is therefore:

$$p(y_k|x) - p_{S\setminus\{i\}}(y_k|x). \quad (2)$$

If this difference is large then the i-th variable is important. If it is small then the variable is less important. The sign of the difference reveals whether the value contributes towards or against class value y_k. This approach was extended in (24) to use log-odds ratios (or weight of evidence), or information gain instead of the difference in predicted class probabilities.

The lack of information about A_i in $p_{S\setminus\{i\}}(y_k|x)$ is approximated with several predictions. For nominal attributes, we replace the actual value of A_i in each prediction with each of the possible values of attribute A_i, and weight each prediction with the prior probability of the value:

$$p(y_k|x \setminus A_i) \doteq \sum_{s=1}^{m_i} p(A_i = a_s) p(y_k|x \leftarrow A_i = a_s) \quad (3)$$

Here m_i is the number of nominal values of attribute A_i and the term $p(y_k|x \leftarrow A_i = a_s)$ represents the predicted probability of y_k when in instance x we replace the actual value of A_i with a_s. For numerical attributes, we use discretization to split the values of numerical attribute A_i into intervals. The middle points of these intervals are taken as the representative replacement values in Eq. (3), for which we compute predictions $p(y_k|x \leftarrow A_i = a_s)$. Instead of prior probabilities of individual values $p(A_i = a_s)$, we use probabilities of the intervals for weighting the predictions.

To demonstrate the behavior of the method an example of an explanation is given. Let a binary domain contain three important (A_1, A_2, and A_3) and one irrelevant attribute (A_4), so the set of attributes is $S = \{1, 2, 3, 4\}$. The class variable C is expressed as the parity (xor) relation of three attributes $C = A_1 \oplus A_2 \oplus A_3$.

Let us assume that we trained a perfect model for this problem. Our correct model classifies an instance $x = (A_1 = 1, A_2 = 0, A_3 = 1, A_4 = 1)$ to class $C = 0$, and assign it the probability $p(C = 0|x) = 1$. When explaining classification for this particular instance $p(C = 0|x)$, method EXPLAIN simulates the lack of knowledge of a single attribute at a time, so it has to estimate $p_{S\setminus\{1\}}(C = 0|x)$, $p_{S\setminus\{2\}}(C = 0|x)$, $p_{S\setminus\{3\}}(C = 0|x)$, and $p_{S\setminus\{4\}}(C = 0|x)$. Without the knowledge about the values of each of the attributes A_1, A_2, and A_3, the model cannot correctly determine the class value, so the correct estimates of class probabilities are $p_{S\setminus\{1\}}(C = 0|x) = p_{S\setminus\{2\}}(C = 0|x) = p_{S\setminus\{3\}}(C = 0|x) = 0.5$ The differences of probabilities $p_S(y_k|x) - p_{S\setminus\{i\}}(y_k|x)$ therefore equal 0.5 for each of the three important attributes, which indicate that these attributes have positive impact on classification to class 0 for the particular instance x. The irrelevant attribute A_4 does not influence the classification, so the classification probability remain unchanged $p_{S\setminus\{4\}}(C = 0|x) = 1$. The difference of probabilities $p_S(C = 0|x) - p_{S\setminus\{4\}}(C = 0|x) = 0$ so the explanation of the irrelevant attribute's impact is zero.

In reality, the trained models are rarely perfect, so the obtained probabilities used in Eq. (2) contain a certain amount of error which translates to an error of explanations. The empirical evaluation (24) has shown that better models produce better explanations.

3.2 IME, all-subsets approach

The one-variable-at-a-time approach is simple and computationally less intensive but has some disadvantages. The main disadvantage is that disjunctive concepts or redundancies between input variables may result in unintuitive con-

tributions for variables (31). A solution was proposed in (30), where all subsets of values are observed. Such procedure demands 2^a steps, where a is the number of attributes and results in the exponential time complexity. However, the contribution of the variable corresponds to the Shapley value for the coalitional game of a players. This allows an efficient approximation based on sampling.

As sampling takes values for attribute A_i from the existing set of values, we don't need any approximation similar to Eq. (3) for numerical attributes in IME.

3.3 Presenting explanations

The working and practical utility of the one-variable-at-a-time contributions and their visualization are illustrated on the well-known Titanic data set. The task is to classify the survival of a passenger in the disaster of the Titanic ship. The three input variables report the passengers' status during travel (first, second, third class, or crew), age (adult or child), and gender. Note the similarity of the problem with many business decision problems, such as churn prediction, mail response, insurance fraud, etc. As an example, random forest (rf) classifier is used. This model is robust and usually provides good prediction accuracy but it is incomprehensible. The Fig. 1a shows an example of an instance explanation for the prediction of the instance with id 2 (a first class adult male passenger). The text at the top includes the class value in question, the instance id, and the type of model. At the bottom, the description contains the type of explanation technique used, the model's prediction for the selected class value, and the actual correct class value for the instance. The input variables' names are shown on the left-hand side and their values for the particular instance are on the right-hand side. The thick dark shaded bars indicate the contributions of the instance's values for each corresponding input variable towards (or against) the class value "survived=yes". The thinner and lighter bars above indicate average contributions of these values across all instances. For the given instance one can observe that "sex=male" speaks against survival and "status=first class" speaks in favor of survival while being an adult has little influence. Thinner average bars above them reveal that being male can be both favorable and dangerous while being in the first class is on average even more beneficial than in the selected case. Note that the same visualization can be used even if some other classification method is applied. A more general view of the model is provided by averaging the explanations for the training data and their visualization in a summary form, which shows the average importance of each input variable and its values. For numerical attributes, explanations for intervals of values are shown; to get sensible intervals, numerical attributes are discretized. An example of such a model explanation for Titanic data set is presented in Fig. 1b. On the left-hand side, the input variables and their values are shown. For each value, the average negative and the average positive contributions across all instances is displayed. Note that negative and positive contributions would cancel each other out if summed together, so it is important to keep them separate. The lighter bars shown are equivalent to the lighter bars in the instance explanation on Fig. 1a. For each input variable, the average positive and negative contributions for all values and instances are shown (darker bars). The visualization reveals that the sex has the strongest effect in random forest model. Traveling in the first or second class has a predominantly positive contribution towards the survival, being a child or female has greater positive than negative contribution, while traveling in the third class has a negative contribution.

4 Implementation in R package ExplainPrediction

The methods EXPLAIN and IME are implemented in the R package ExplainPrediction. The top level entry is the explainVis function which explains predictions of a given model and visualizes the explanations. In this section, we explain the parameters of explainVis and show how to call it. We also share some useful tips for using the explanations.

4.1 Controlling explanations

The function explainVis enables fine control over computation and visualization of explanations through its arguments listed in Listing 1 and explained below.

Parameters controlling input/output

model specifies the input prediction model.

trainData is the input data set which is used to compute average explanations, discretization, and other information needed for explanation of instances and model.

testData is the input data set containing instances that are going to be explained.

fileType determines the graphical format of the output visualization file: pdf, eps, emf, jpg, png, bmp, or tif. If "none" is specified, the visualization is directed to a graphical window.

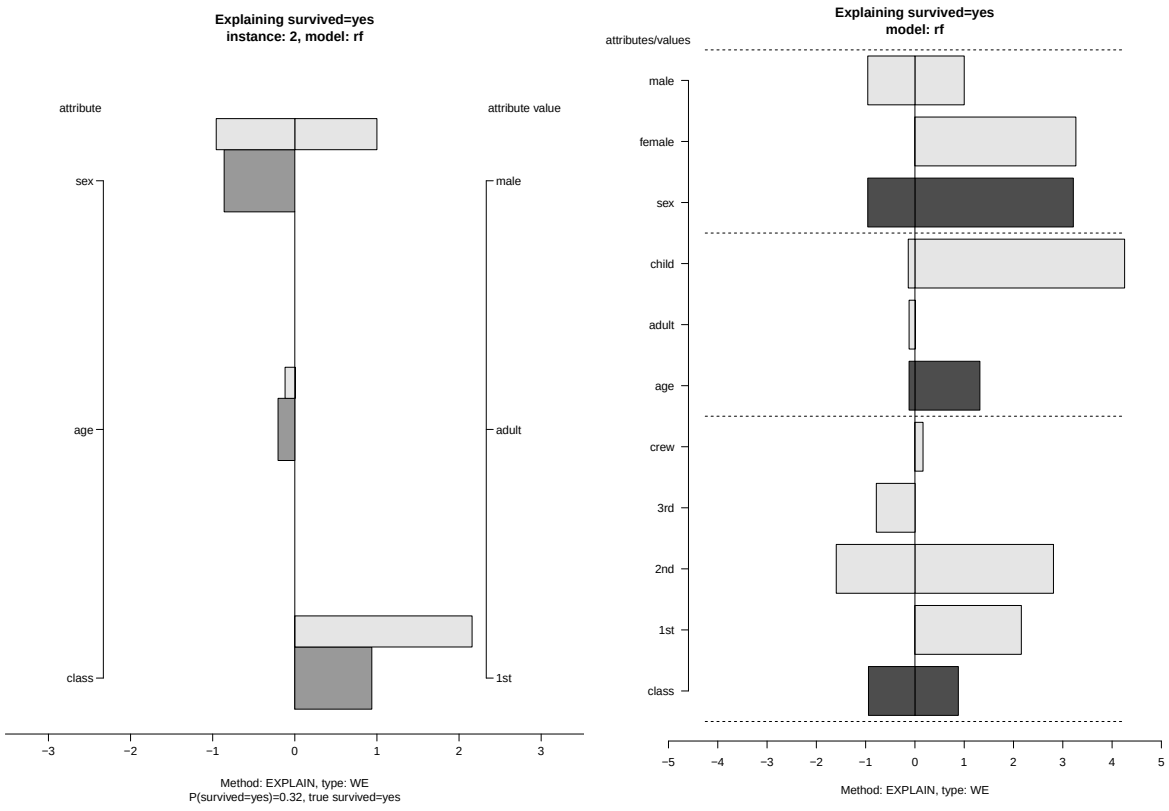
dirName specifies the output folder where resulting visualization files will be saved.

fileName contains the file name of the resulting output visualization files.

The parameters of both explanation methods

method specifies the explanation method, either EXPLAIN or IME.

classValue specifies the class value for which explanations are generated.



a) b)

Figure 1: An instance explanation a) and a model explanation b) for the random forest model classifying the Titanic data set. The tiny bars in the instance explanation represent the average positive and average negative contributions of the values and are equal to the corresponding value-bars in the model explanation (note the difference in scale).

visLevel determines the level of explanations desired, i.e. the model level or instance level explanations.

estimator specifies the feature evaluation method used to greedily discretize attributes needed when averaging explanation over intervals of numeric attributes. The default value NULL invokes discretization with attribute evaluation algorithms ReliefF (classification) or RReliefF (regression) from R package CORElearn (25).

recall can provide the list with all explanations data returned by one of the previous calls to function explainVis, which speeds-up the computations.

Parameters specific to EXPLAIN (see (24))

explainType specifies for the EXPLAIN method how the prediction with knowledge about given feature and the prediction without knowledge of this feature are combined into the final explanation. The values "WE", "infGain", and "predDiff" mean that the difference is interpreted as the weight of evidence, information gain, or plain difference of predictions, respectively. For regression problem only the difference of predictions is available.

naMode specifies for the EXPLAIN method how the impact of missing information about certain feature value is estimated. It can be estimated by the weighted average of predictions over all possible feature's values, or by inserting NA value as a feature value.

nLaplace specifies for the EXPLAIN method and classification problems the value to be used in Laplace correction of estimated probabilities.

Parameters specific to IME (see (30))

pError specifies for the IME method the estimated probability of an error in explanations. Together with the *err* parameter, this determines the number of needed samples.

batchSize specifies for the IME method the number of samples processed for each explanation in one batch. To reduce processing overhead in calls to *predict* function we process several samples at once. This strategy reduces the overhead but may process more samples than required.

maxIter sets the maximal number of iterations in IME method allowed for a single explanation.

Listing 1: Top level call to explanation methods and their visualization.

```
explainVis(model, trainData, testData, method=c("EXPLAIN", "IME"), classValue=1,
    fileType=c("none","pdf","eps","emf","jpg","png","bmp","tif","tiff"), dirName=getwd(), fileName="explainVis",
    visLevel=c("both","model","instance"), explainType=c("WE","infGain","predDiff"), naMode=c("avg", "na"),
    nLaplace=nrow(trainData), estimator=NULL, pError=0.05, err=0.05, batchSize=40, maxIter=1000,
    genType=c("rf", "rbf", "indAttr"), noAvgBins=20, displayAttributes=NULL, modelVisCompact=FALSE,
    displayThreshold=0.0, colors=c("navyblue", "darkred", "blue", "red", "lightblue", "orange"),
    normalizeTo=0, noDecimalsInValueName=2, recall=NULL
    modelTitle=ifelse(model$noClasses==0,"Explaining \%R\nmodel: \%M", "Explaining \%R=\%V\\nmodel: \%M"),
    modelSubtitle="Method: \%E, type: \%X",
    instanceTitle=ifelse(model$noClasses==0, "Explaining \%R\\ninstance: \%I, model: \%M",
                                             "Explaining \%R=\%V\\ninstance: \%I, model: \%M"),
    instanceSubtitle=ifelse(model$noClasses==0, "Method: \%E\\nf(\%I)=\%P, true \%R=\%T",
                                                "Method: \%E, type: \%X\\nP(\%R=\%V)=\%P, true \%R=\%T") )
```

genType specifies the type of data generator used to generate random part of instances in method IME. The generators from R package semiArtificial(22) are used: "rf" stands for the random forest based generator, "rbf" invokes RBF network based generator, and indAttr" assumes independent attributes and generates values for each attribute independently.

noAvgBins specifies for the IME method the number of discretization bins used to present model level explanations and average explanations.

Visualization parameters:

displayAttributes is the vector of attribute names which are visualized in model level visualization.

modelVisCompact determines if attribute values are displayed in model level visualization.

displayThreshold specifies the threshold value for absolute values of explanations below which feature contributions are not displayed in instance and model explanation graphs.

normalizeTo determines the value for instance level visualization to which the sum of the absolute values of feature contributions are normalized (e.g., 1 or 100).

colors determines colors used in visualization.

noDecimalsInValueName specifies how many decimal places will numeric feature values use in visualizations.

modelTitle, *modelSubtitle*, *instanceTitle*, and *instanceSubtitle* are string templates for various titles of different graphs. The template uses several variables, which are inserted at the appropriate place: response variable %R, the selected class value for explanation %V, type of model %M, explanation method %E, explanation type %X, instance name %I, predicted value/probability of the response %P, and the true value of the response %T.

4.2 Producing explanations

The function explainVis generates explanations and their visualizations given the trained model, its training data, and data for which we want explanations. This is the front-end explanation function which takes care of everything, internally calling other functions. The produced visualizations are output to a graphical device or saved to a file. The function returns a list with explanations, average explanations, and additional data like discretization used and data generator. An example of a call is presented in Listing 2.

The explanations support several models implemented in packages CORElearn, randomForest, nnet, and e1071. Adding support for new predictors is easy and involves preparation of class names and class values in the format expected by the package ExplainPrediction when calling the predictor. This is demonstrated in the function wrap4Explanation, which is part of the ExplainPrediction package.

4.3 Tips for using the explanations

The presented explanation techniques have many successful applications (shortly reviewed in Section 2). Here we present a few tips for successful practical use of explanations.

For many real-world problems gaining the trust of users is essential to assure successful application of machine learning models. Instance and model explanation can serve as convenient ice-breakers. If a user can check for some instances that the generated explanations match his/her understanding of the problem, this greatly increases chances of success and is more convincing than reporting high classification accuracy. This is true even for mispredicted instances as long as the model's reasoning is sensible for users.

For larger data sets with many attributes, time to produce explanations with the IME method can be substantial. However, in spite of theoretical advantages of IME over EXPLAIN, in practice, these two methods mostly produce similar explanations. This indicates that in real-world pro-

Listing 2: A code that generates explanations of model and instances.

```
require(ExplainPrediction)
require(CORElearn)
# use iris data set, split it randomly into a training and testing set
trainIdxs <- sample(x=nrow(iris), size=0.7*nrow(iris), replace=FALSE)
testIdxs <- c(1:nrow(iris))[-trainIdxs]
# build random forests model with certain parameters
modelRF <- CoreModel(Species ~ ., iris[trainIdxs,], model="rf", rfNoTrees=100, selectionEstimator="MDL",
                     minNodeWeightRF=5)
# generate model and instance explanations and visualize them in a graphical window
explainVis(modelRF, iris[trainIdxs,], iris[testIdxs,], method="EXPLAIN", fileType="none", naMode="avg",
           explainType="WE", classValue=1)
```

blems there are few redundant attributes and even less redundant attributes of exactly the same strength (if redundant attributes are not of the same strength, learning selects the stronger ones and there is no redundancy in the model). In practice, we can compare the behavior of EXPLAIN and IME on a subsample of instances and attributes. If explanations are similar, the EXPLAIN method can be used instead of IME.

To reach a desired graphical design (e.g., colors and headings) and show only the most impactful attributes requires some tweaking of visualization parameters. To avoid regeneration of explanations for each user interaction with explanations, we provide the *recall* parameter. In the first call to the explainVis function, we have to store the invisibly returned list to a variable and supply this variable as the value of parameter *recall* in subsequent calls to explainVis. In this case the function reuses already computed explanations, average explanations, discretization, etc., and only display data differently according to supplied input/output and visualization parameters (visLevel, dirName, fileType, displayAttributes, modelVisCompact, displayThreshold, normalizeTo, colors, noDecimalsInValueName, modelTitle, modelSubtitle, instanceTitle, and instanceSubtitle). Using this hint can make user interactions with explanations instantaneous even for large data sets.

5 Conclusions

We presented two general methods for explanation of prediction models and their implementation in the ExplainPrediction package. The methods allow explanation of individual decisions as well as the prediction model as a whole. The explanations provide information on how the individual input variables influence the outcome of prediction models, thus improving their transparency and comprehensibility. The general methods allow users to compare different types of models or replace their existing model without having to replace the explanation method. The explanation methods can be efficiently computed and visualized, and their implementation offers several parameters that control the speed and precision of the computed explanations, convergence rate and visualization of explanations. Several models are supported and adding support in almost any prediction model is easy.

The simplicity and elegance of the perturbation based explanations coupled with efficient implementations and visualization of instance- and model-based explanations allow application of general explanation approaches to new areas. We expect that broader practical use will spur additional research into explanation mechanisms and improvements in the visual design of explanations. There are also many possibilities for methodological improvements. An idea worth pursuing seems integration of game theory based sampling and formulation of explanations as an optimization problem. The implementation of IME in the ExplainPrediction package could be improved by rewriting it in C language and using better, context-dependent, sampling method.

Acknowledgment

We acknowledge the support of the Slovenian Research Agency, ARRS, through research programme P2-0209 (Artificial Intelligence and Intelligent Systems) and projects J6-8256 (New grammar of contemporary standard Slovene: sources and methods) and L1-7542 (Advancement of computationally intensive methods for efficient modern general-purpose statistical analysis and inference). Fruitful discussions with Erik Štrumbelj improved the implementation of the IME method.

Literature

[1] Charu C Aggarwal, Chen Chen, and Jiawei Han. The inverse classification problem. *Journal of Computer Science and Technology*, 25(3):458–468, 2010.

[2] Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. What is relevant in a text document?: An interpretable machine learning approach. *PloS ONE*, 12(8):e0181142, 2017.

[3] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and

Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS ONE*, 10(7):e0130140, 2015.

[4] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11 (Jun):1803–1831, 2010.

[5] David Barbella, Sami Benzaid, Janara M Christensen, Bret Jackson, X Victor Qin, and David R Musicant. Understanding support vector machine classifications via a recommender system-like approach. In R. Stahlbock, S. F. Crone, and S. Lessmann, editors, *Proceedings of International Conference on Data Mining*, pages 305–311, 2009.

[6] Adriano Barbosa, FV Paulovich, Afonso Paiva, Siome Goldenstein, Fabiano Petronetto, and LG Nonato. Visualizing and interacting with kernelized data. *IEEE transactions on visualization and computer graphics*, 22(3):1314–1325, 2016.

[7] Marko Bohanec, Mirjana Borštnar Kljajić, and Marko Robnik-Šikonja. Explaining machine learning models in sales predictions. *Expert Systems with Applications*, 71:416–428, 2017.

[8] Marko Bohanec, Marko Robnik-Šikonja, and Mirjana Kljajić Borštnar. Decision-making framework with double-loop learning through interpretable black-box machine learning models. *Industrial Management & Data Systems*, 117(7):1389–1406, 2017.

[9] Zoran Bosnić, Jaka Demšar, Grega Kešpret, Pedro Pereira Rodrigues, João Gama, and Igor Kononenko. Enhancing data stream predictions with reliability estimators and explanation. *Engineering Applications of Artificial Intelligence*, 34:178–192, 2014.

[10] Jaka Demšar and Zoran Bosnić. Detecting concept drift in data streams using model explanation. *Expert Systems with Applications*, 92:546 – 559, 2018.

[11] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 24(1):44–65, 2015.

[12] Lutz Hamel. Visualization of support vector machines with unsupervised learning. In *Proceedings of 2006 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, 2006.

[13] Katja Hansen, David Baehrens, Timon Schroeter, Matthias Rupp, and Klaus-Robert Müller. Visual interpretation of kernel-based prediction models. *Molecular Informatics*, 30(9):817–826, 2011.

[14] Aleks Jakulin, Martin Možina, Janez Demšar, Ivan Bratko, and Blaž Zupan. Nomograms for visualizing support vector machines. In Robert Grossman, Roberto Bayardo, and Kristin P. Bennett, editors, *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 108–117. ACM, 2005.

[15] Vincent Lemaire, Raphael Féraud, and Nicolas Voisine. Contact personalization using a score understanding method. In *Proceedings of International Joint Conference on Neural Networks (IJCNN)*, 2008.

[16] David Martens and Foster Provost. Explaining documents' classifications. Technical report, Center for Digital Economy Research, New York University, Stern School of Business, 2011. Working paper CeDER-11-01.

[17] David Meyer, Friedrich Leisch, and Kurt Hornik. The support vector machine under test. *Neurocomputing*, 55:169–186, 2003.

[18] Rok Piltaver, Mitja Luštrek, Matjaž Gams, and Sanda Martinčić-Ipšić. What makes classification trees comprehensible? *Expert Systems with Applications*, 62: 333–346, 2016.

[19] Brett Poulin, Roman Eisner, Duane Szafron, Paul Lu, Russell Greiner, David S. Wishart, Alona Fyshe, Brandon Pearcy, Cam Macdonell, and John Anvik. Visual explanation of evidence with additive classifiers. In *Proceedings of AAAI'06*. AAAI Press, 2006.

[20] Marko Pregeljc, Erik Štrumbelj, Miran Mihelčič, and Igor Kononenko. Learning and explaining the impact of enterprises' organizational quality on their economic results. In R. Magdalena-Benedito, M. Martinez-Sober, J. M. Martinez-Martinez, P. Escandell-Moreno, and J. Vila-Frances, editors, *Intelligent Data Analysis for Real-Life Applications: Theory and Practice*, pages 228–248. Information Science Reference, IGI Global, 2012.

[21] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016.

[22] Marko Robnik-Šikonja. Data generators for learning systems based on rbf networks. *IEEE Transactions on Neural Networks and Learning Systems*, 27(5):926–938, May 2016.

[23] Marko Robnik-Šikonja. *ExplainPrediction: Explanation of Predictions for Classification and Regression*, 2017. URL `http://cran.r-project.org/package=ExplainPrediction`. R package version 1.3.0.

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

[25] Marko Robnik-Šikonja and Petr Savicky. *CORElearn - classification, regression, feature evaluation and ordinal evaluation*, 2017. URL `http://cran.r-project.org/package=CORElearn`. R package version 1.52.0.

[26] Andrea Saltelli, Karen Chan, and E. Marian Scott. *Sensitivity analysis*. Wiley, Chichester; New York, 2000.

[27] Alexander Schulz, Andrej Gisbrecht, and Barbara Hammer. Using discriminative dimensionality reduction to visualize classifiers. *Neural Processing Letters*, 42(1):27–54, 2015.

[28] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[29] Erik Štrumbelj, Zoran Bosnić, Igor Kononenko, Branko Zakotnik, and Cvetka Grašič Kuhar. Explanation and reliability of prediction models: the case of breast cancer recurrence. *Knowledge and information systems*, 24(2):305–324, 2010.

[30] Erik Štrumbelj and Igor Kononenko. An Efficient Explanation of Individual Classifications using Game Theory. *Journal of Machine Learning Research*, 11: 1–18, 2010.

[31] Erik Štrumbelj, Igor Kononenko, and Marko Robnik-Šikonja. Explaining instance classifications with interactions of subsets of feature values. *Data & Knowledge Engineering*, 68(10):886–904, 2009.