SN Computer Science (2024) 5:563
https://doi.org/10.1007/s42979-024-02892-4

ORIGINAL RESEARCH

A Reference Process for Assessing the Reliability of Predictive Analytics Results

Simon Staudinger[1] · Christoph G. Schuetz[1] · Michael Schrefl[1]

Received: 14 February 2022 / Accepted: 10 April 2024

Abstract
Organizations employ data mining to discover patterns in historic data in order to learn predictive models. Depending on the predictive model the predictions may be more or less accurate, raising the question about the reliability of individual predictions. This paper proposes a reference process aligned with the CRISP-DM to enable the assessment of reliability of individual predictions obtained from a predictive model. The reference process describes activities along the different stages of the development process required to establish a reliability assessment approach for a predictive model. The paper then presents in more detail two specific approaches for reliability assessment: perturbation of input cases and local quality measures. Furthermore, this paper describes elements of a knowledge graph to capture important metadata about the development process and training data. The knowledge graph serves to properly configure and employ the reliability assessment approaches.

Keywords Business intelligence · Business analytics · Decision support systems · Data mining · CRISP-DM

Introduction

Organizations employ data mining to discover patterns in historic data in order to learn predictive models. The CRoss-Industry Standard Process for Data Mining (CRISP-DM) [1] provides organizations with guidelines for the application of data mining to reach the defined business goals. The CRISP-DM includes six stages: (i) business understanding, (ii) data understanding, (iii) data preparation, (iv) modeling, (v) evaluation, and (vi) deployment. As a generic process model, the CRISP-DM is used for data mining projects in various domains [2–4]. Once deployed, the developed predictive models are used by decision-makers to obtain predictions for individual cases and act accordingly.

Depending on the predictive model, the predictions may be more or less accurate, raising the question about the reliability of individual predictions. Consider, for example, a classification model that allows a bank employee to decide whether a specific client will default on a requested loan in the future. If similar cases from the past cannot be clearly associated with a specific outcome, or similar cases have not been present in the input data, the prediction will not be reliable. Another example is the use of a clustering model to collect existing customers into customer groups. If a customer is near the border of two clusters the assignment of that customer to a customer group may not be reliable. Regardless, predictive models will come up with a prediction. The analyst charged with taking a decision, however, should keep in mind potential issues with reliability when acting on a prediction. Actions based on unreliable predictions may lead to potentially costly failures and missed business opportunities.

In this paper we propose a reference process aligned with the CRISP-DM to enable the assessment of reliability of individual predictions obtained from a predictive model. We illustrate the reference process using the case of classification over a real-world dataset in telemarketing for the

This article is part of the topical collection "Data Science, Technology and Applications" guest edited by Slimane Hammoudi and Christoph Quix.

✉ Simon Staudinger
simon.staudinger@jku.at

Christoph G. Schuetz
christoph.schuetz@jku.at

Michael Schrefl
michael.schrefl@jku.at

[1] Institute of Business Informatics-Data and Knowledge Engineering, Johannes Kepler University Linz, Altenberger Straße 69, Linz 4040, Austria

Published online: 18 May 2024

banking business [5]. We argue that in order to be able to accurately assess the reliability of individual predictions an analyst must consider the actual input data, the data used for training the predictive model, and the specific procedures regarding collection and preparation of the data. Thus, the reference process defines activities along the entire CRISP-DM life cycle and defines the knowledge about the data mining process that must be recorded and considered during each process stage.

During the business understanding stage, the development team determines the methods employed for the assessment of the reliability of predictions; we refer to these methods as *reliability assessment approaches* throughout this paper. The development team may select predefined methods, for example, the perturbation of input cases or the use of local quality measures, or the development team may develop new approaches. During the data understanding and data preparation stages, the development team documents the activities and design decisions regarding data gathering, data cleaning, and data formatting, which serves to configure the reliability assessment approaches during the modeling and evaluation stages. For example, information about the precision of a sensor may inform the application of the *perturbation approach*. With the deployment of the model in day-to-day business, the reliability assessment approaches selected or developed alongside the predictive model are also deployed. The reliability assessment approaches can be used by analysts to better assess the reliability of an individual prediction for a new input case. In this paper we present in more detail two specific approaches for reliability assessment: perturbation of input cases and local quality measures.

Inspired by the field of numerical analysis, where the condition number describes the magnitude of change of a function's output in connection with changes of the function's input [6], the perturbation of input cases for individual predictions aims to uncover the effect of alterations of an input case's feature values on the outcome of a prediction. If small alterations of feature values would lead to a completely different prediction then the reliability of that prediction is questionable. For example, consider a predictive model predicting whether a bank customer will repay a requested loan based on a number of input features such as income, marital status, and education. If a minor change in the monthly income, e.g., 50 euros, leads to a changed prediction of the model, the prediction may not be reliable since a small change in income may happen at any time or a person's exact income may vary from month to month.

While global measures of a predictive model's overall performance, e.g., accuracy, are often the only guidance regarding the reliability of individual predictions, local quality measures of a predictive model's performance for a particular feature space may complement the insights offered by global measures. Individual cases in different areas of the data space may lead to different performance of the predictive model, for example, due to the distribution of the training data in the feature space. A new data point located in a densely populated part of the feature space may obtain a more reliable prediction than a new data point in a sparsely populated part. The global measures would be the same for both predictions, since the global measures describe the predictive model's overall performance. Related work likewise investigates the the use of local metrics [7–10], demonstrating that it can be fruitful to collect model metrics not only on a global level but also in the local area of interest of the data. Stating a local quality measure additionally to the global measure gives analysts a better impression of the reliability of an individual prediction.

At each stage the proposed reference process requires the development team to document design decisions and capture important metadata in a knowledge graph. The knowledge graph is a machine-readable representation of the activities, entities, and agents of the reference processes at a generic, method-specific, and problem-specific level for the purposes of recommendation, quality management, and auditability. The knowledge graph is based on the W3C recommendation for a *provenance ontology* (PROV-O) [11]. Other work has employed PROV-O as the basis to improve explainability of automated decisions in order to meet regulatory requirements [12].

The main contribution of this paper is as follows. The proposed reference process provides a structured approach to assess the reliability of individual prediction results along the CRISP-DM life cycle. Instead of relying solely on global indicators of model reliability, e.g., accuracy or precision, which are indifferent to the individual case that a prediction shall be obtained for and, therefore, do not account for differences between input cases, an analyst may assess the reliability of an individual case by using information gathered along the various stages of the CRISP-DM. We illustrate execution of the reference process by detailing two assessment approaches for classification problems, namely the use of local quality measures and the perturbation of input cases. Nevertheless, the presented reference process can be extended with additional assessment approaches and adapted for the use of other prediction methods.

This paper is an extended version of a previous conference paper [13]. In particular, this paper generalizes the previously proposed process for reliability assessment in the context of classification, describing a reference process at a more abstract level, which can be adapted for other prediction methods. Furthermore, this paper introduces a knowledge graph that captures knowledge required for enabling the assessment of reliability of predictions.

The remainder of this paper is organized as follows. In "Related Work" we review related work. In "Overview" we give an overview of the proposed reference process and the

knowledge graph. In "Business Understanding" we describe the business understanding stage. In "Data Understanding" we describe the data understanding stage. In "Data Preparation" we describe the data preparation stage. In "Modeling and Evaluation" we describe the modeling and evaluation stages, including the proposed approaches of perturbation of input cases and the use of local quality measures. In "Deployment" we discuss the deployment stage. In "Conclusions" we conclude the paper.

Related Work

A white paper published by the Technical Inspection Association (TÜV) Austria in cooperation with researchers from Johannes Kepler University Linz highlights the importance of reliance in artificial intelligence applications [14]. The white paper proposes the idea of certification of machine learning systems to avoid hazardous consequences due to incorrect usage of these systems. First steps towards a certification procedure for machine learning applications are aimed to foster the development of reliable artificial intelligence applications.

Wing [15] presents key insights regarding trustworthy artificial intelligence. Reliability is a core property of trustworthy artificial intelligence, besides other important properties, e.g., safety, security, robustness, and fairness. The application of formal methods in combination with artificial intelligence systems employing trained models, e.g., neural networks, which make predictions based on statistical probabilities, is challenging due to the probabilistic nature of the predictive models. Furthermore, machine learning models are built with respect to a given data set which must not contain all possible real-world cases but is further used to predict previously unseen cases. Wing also mentions that more research towards trustworthy artificial intelligence and, in this context, towards reliable artificial intelligence will be necessary.

The *perturbation approach* presented in this paper is related to metamorphic software testing [16–18]. Metamorphic software testing can be used to deal with the *oracle problem*, which occurs when the the correct behavior of the software cannot be distinguished from an incorrect behavior due to missing formal specifications or assertions. Metamorphic relations are defined, which describe the differences between input and output of a software. If the real outputs differ from the expected ones, it is a sign that there may be an error or bug in the software.

In metamorphic testing, the output of the software can be used as a follow-up test case serving as new input. With this technique, a possibly infinite amount of test cases for the software can be created. An example of a metamorphic relation is the case of two search queries, where the second query is a restriction of the first query, e.g., by adding an additional condition in the WHERE clause of an SQL query. If during metamorphic testing the result of the second query is not a subset of the result of the first query, the formulation of the query is wrong. The biggest challenge in metamorphic testing is the definition of the metamorphic relations suitable for the software under investigation. Other research also examined the use of metamorphic relations to ensure the soundness of machine-learning classifiers [19–21].

Dynamic classifier selection (DCS) is used in various domains to find the most suitable classifier from a group of different classifiers trained for the same classification problem [22]. Given multiple classifiers for the same problem, there are various ways to come to the final prediction, for example, majority voting, where the final outcome is the most frequently predicted class among all classifiers. DCS follows the approach of finding the best-fitting classifier for a given input case and using the prediction made by that best-fitting classifier as the finally predicted outcome. There are different ways described to select the best-fitting classifier, for example, through the use of local regions within the feature space to examine a classifier's performance in the area of the new input case [10, 22, 23].

Besides introducing a generalization of the previously published process [13] as well as a knowledge graph, we integrate in the proposed reference process in this paper aspects from related work as follows. During metamorphic testing a test engineer uses systematically altered input cases in combination with a metamorphic relation to ensure the correct behaviour of a software. In this paper we use slightly altered input cases within the *perturbation approach* to investigate whether small input changes influence the prediction of a model. DCS is mainly about finding the best suitable classifier from a group of classifiers, for example, by use of local measures. DCS does not consider using local measures to evaluate the reliability of individual predictions made by the same classifier. We use the local-area idea within our *local measure approach* to examine differences between global and case specific local measures.

Overview

This section first gives an overview of the proposed reference process for facilitating assessment of reliability of predictive analytics results before describing the different abstraction levels of metadata that, collected into a knowledge graph, facilitate the application of the reference process in different data mining projects.

We propose the use of three abstraction levels for the reference process, namely a generic, a method-specific, and an instance-specific level. The generic level concerns activities and entities at each stage of the CRISP-DM, independent of

the type of data mining problem at hand, e.g., classification or clustering. The method-specific level defines specializations of the generic activities and entities at each stage of the CRISP-DM that are specific to individual methods, for example, classification requires the determination of the scale of a feature during the data understanding stage. The problem-specific level then represents instances of the method-specific activities and entities filled with concrete values of an individual prediction case, for example, determining the scale of the feature *education* for the problem of predicting whether a new customer will likely be able to pay back a requested loan.

Reference Process

Figure 1 illustrates the proposed reference process, which aligns with the six stages of the CRISP-DM [1]. Hence, in order to allow for the assessment of reliability of individual predictions in productive use, the development team must conduct appropriate activities along every stage of the data mining project. The proposed reference process can then be refined for different types of data mining projects and adapted to the peculiarities of each individual project. For example, Fig. 2 shows activities for enabling the reliability of classification results. Throughout this paper we use *classification* as a running example for illustrating the stages of the proposed reference process.

In the *business understanding stage* of a data mining project, the development team must choose the appropriate approach for reliability assessment, which must be in line with the elicited business goals in general and the data mining goals in particular. For example, the choice of assessment approach may fall on the *perturbation of test cases*, which consists of alternating the values of sensitive independent features and witness changes in the prediction (see "Perturbation of Input Cases").

Business understanding is closely interrelated with the *data understanding* stage, which consists of the elicitation and examination of the data required to achieve the business and data mining goals. In order to enable assessment of reliability, the development team must closely examine the characteristics of the data and document the findings. Metadata about the input features provides important information for development decisions regarding reliability assessment at later stages. Among the collected metadata are the scale of a feature (or level of measurement), i.e., nominal, cardinal, or ordinal, the volatility of measured values, e.g., the accuracy of sensor data, or existing restrictions on values, e.g., allowed range of feature values. There will typically be a feedback loop between business understanding and data understanding. For example, the choice of assessment approach during the business understanding stage may often only be made after examining the characteristics of the data

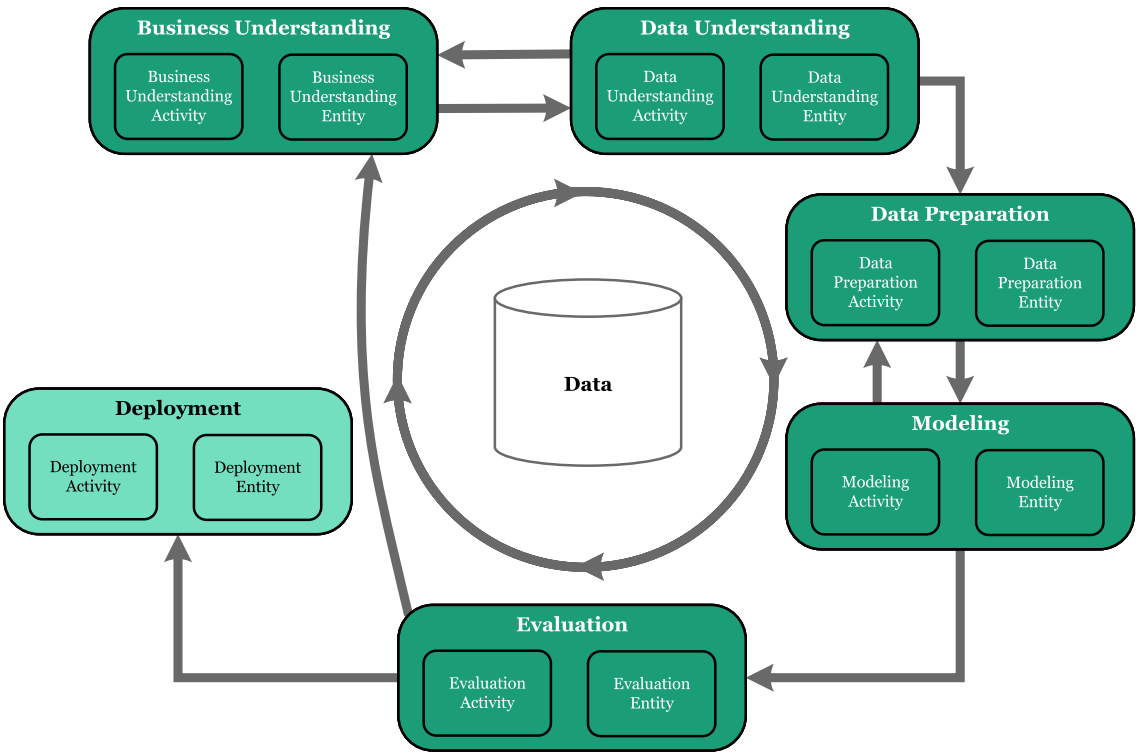

Fig. 1 Reference process for facilitating assessment of reliability of predictive analytics results along the CRISP-DM stages

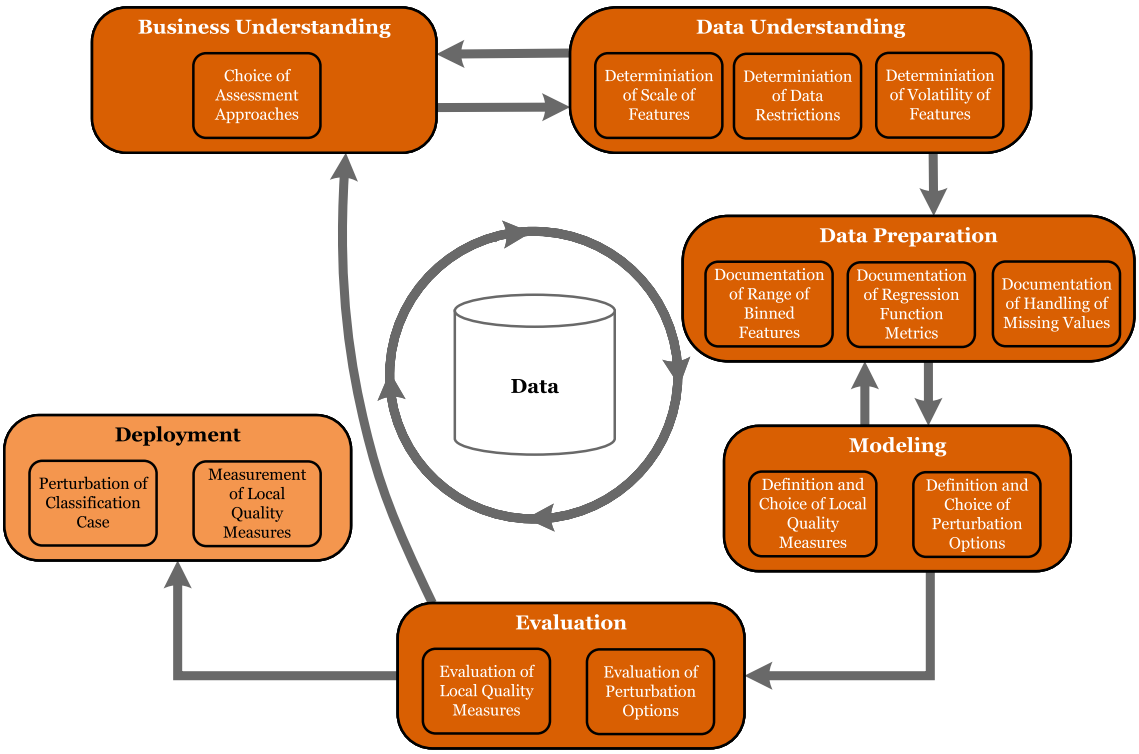

Fig. 2 Example activities for enabling assessment of reliability of classification results [13]

Data cleaning and data transformation in the *data preparation* stage, which is essential for using the collected data to train a predictive model, may exercise considerable influence on the reliability of the predictions; the data cleaning and data transformation steps should be documented. For example, using regression to fill missing values or binning feature values into different categories, while often necessary to improve the quality of the data and render those data usable for the analysis, may also affect the reliability of individual predictions in productive use. Information about such activities, if documented, may inform the analyst when assessing the reliability of a prediction. For example, a change of the outcome of a prediction when varying an input value of the predictor within the boundaries of a bin hints at an unreliable prediction (see "Data Preparation").

The *modeling* stage comprises the selection of the algorithm as well as the definition of the test strategy, e.g., cross-validation, which is employed in the following *evaluation* stage. In the modeling stage the development team must decide on the specific configuration of the previously selected approaches for reliability assessment, which can be used in the evaluation stage as well as after deployment of the developed predictive analytics solution. This paper proposes two approaches for reliability assessment although the process is not limited to those approaches. The first approach is perturbation of input cases, which at the modeling stage requires the configuration of perturbation options and perturbation modes (see "Perturbation of Input Cases"). The second approach is the calculation of local quality measures, which at the modeling stage requires the development team to choose appropriate measures. For selection, development, and configuration of the approaches for reliability assessment, the development team relies on the metadata collected during the previous stages.

Perturbation of input cases consists of systematically altering the values of input cases to find sensitive features where small changes upend the prediction. The perturbation option and the perturbation mode determine how the values of input cases are altered. The appropriateness of a perturbation option for a problem depends on the scale of feature, the volatility of the feature, and possibly other criteria. For example, perturbation may consist of looking at the predictions made by the predictive model when altering an input value within a 10% window; that perturbation option is only appropriate for cardinal values. Looking at all possible perturbations, particularly when multiple features are perturbed, may not be feasible in a reasonable amount of time. The perturbation mode determines which perturbed test cases are presented to the analyst.

The calculation of local quality measures aims at comparing the characteristics of the data space around the input data with the characteristics of the data available during the training of the model. For example, a classifier for credit default may show high accuracy for classifying males aged

between 30 and 45 but low accuracy for women aged 20–25, which means that the classifier will be less accurate when predicting a case with that gender and age range.

Whether a certain approach for reliability assessment can be sensibly employed depends on the prediction method, e.g., classification, which requires further tailoring to the individual problem, e.g., determining the credit rating. Hence, a selected approach for reliability assessment must be fitted to the individual problem. When a selected approach is already employed in the evaluation stage to assess the quality of the trained models, the development team may also evaluate the parameters of the selected approach for reliability assessment, which may lead to the adjustment of certain parameters. For example, the development team may choose to alter the range of the perturbations or the distance around the input values delimiting the local space for calculation of local quality measures.

Finally, during the *deployment* stage, trained and tested predictive models are integrated in day-to-day business, which allows analysts to productively use the developed predictive models for making predictions in individual cases. Likewise, an analyst may then employ the selected approaches for assessment of reliability of individual predictions. The results of the reliability assessment allow an analyst to more confidently decide whether to trust a prediction and take the best decision for the business.

Note that the development team can conduct the activities described in the reference process alongside the development of the predictive model or afterwards. While building the predictive model, the reliability assessment activities can be performed during each phase of the CRISP-DM. If a predictive model has already already been trained and put to use, the activities required to enable reliability assessment have to be performed independently of the development of the predictive model.

Knowledge Graph

At each stage of the proposed reference process the development team should capture in a *knowledge graph* the relevant metadata about the activities conducted by various agents, producing entities that represent the knowledge about the data mining process which can be leveraged for assessment of the reliability of individual predictions. In a sense, the captured knowledge can be considered provenance information documenting the origin of the predictive model, specifically the design decisions that led to the development of the predictive model and the choice of reliability assessment methods. Thus, the proposed knowledge graph builds on the W3C recommendation for representation of provenance information, the provenance ontology PROV-O [11]

The knowledge graph documenting the design decisions regarding the predictive models and reliability assessment comprises three main types of elements, taken from PROV-O: activities, entities, and agents. An entity is any kind of design artifact produced by an activity. An agent can be any person or machine component which is involved in or responsible for the execution of an activity.

The knowledge graph describes knowledge about the data mining process on a generic, method-specific, and problem-specific level, consisting of both ontological knowledge as well as instance data (Fig. 3). Furthermore, at each level, the knowledge graph has a *development* view and a *deployment* view, tracking the knowledge about the development of a predictive model and its deployment, respectively. Elements in the development view serve to assess the reliability of an individual prediction after deployment of the predictive model. Elements in the deployment view represent the actual use of methods for reliability assessment for the purposes of auditability and checking the consistency of the application of the selected methods.

The knowledge graph's generic level describes activities and entities relevant for different problems such as classification and clustering. In particular, the generic level defines the basic vocabulary of the knowledge graph. The method-specific level captures knowledge regarding the application of individual prediction methods, e.g., classification, and the corresponding methods for reliability assessment, e.g., perturbation of input cases. The activities depicted in Fig. 2 represent knowledge at the method-specific level. Finally, the

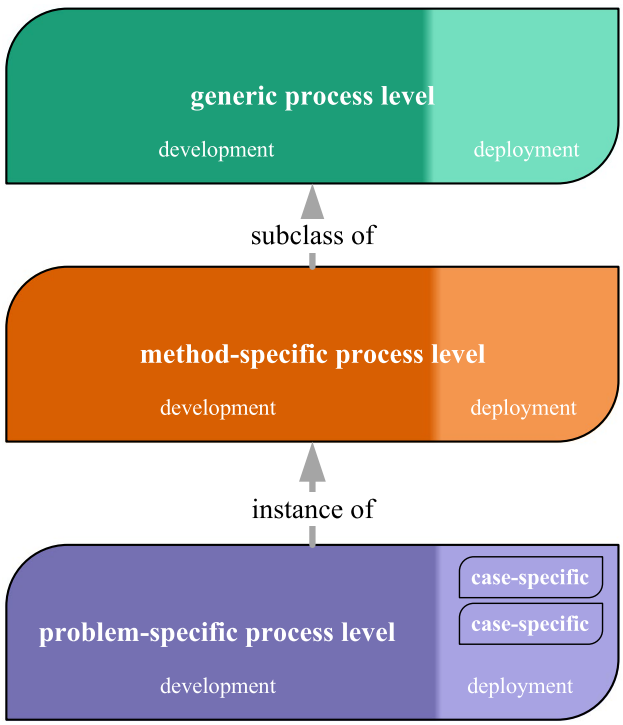

Fig. 3 Abstraction levels of the knowledge graph

problem-specific level describes the development choices for a particular problem, e.g., prediction of credit default. At this level, for example, the specific parameters of the perturbation options are captured. At the problem-specific level, the deployment view also captures case-specific knowledge, e.g., the application of perturbation for predicting the default risk of a particular customer.

We emphasize that the presented reference process, although illustrated and discussed in the context of classification of structured data, is not limited to the problem of classification but potentially also applicable for other prediction problems. Other prediction problems may follow a different method-specific process, which would become another method-specific refinement of the proposed reference process.

During "Business Understanding" – "Deployment" we present examples, on each of the three levels, for activities and entities created during each of the stages. In the following, we want to emphasize that the stages, which are explained in detail in the next sections, are closely connected and dependent on each other, therefore, we provide graph examples which include activities and entities across multiple stages of the CRSIP-DM. In order to keep the explanation of the model comprehensible we omit the agents that are assigned to each activity and entity in most of the examples.

Generic Level

The generic process level is the starting point for the modeling of assessment processes on the method-specific level. Each stage of the CRISP-DM contains a generic activity and a generic entity as shown in Fig. 1, which can be seen as abstract superclass for method-specific subclasses. The generic level provides the connection between the activities and entities within all six stages of the CRISP-DM. Activities and entities included in the generic level are linguistic instances of the PROV-O classes activity and entity, technologically represented as subclasses in RDF. Likewise, the connections between the elements are also linguistic instances of the properties from PROV-O and in our KG technically represented as subproperties.

Each entity that is created during a stage of the CRISP-DM is assigned to an activity related to the same stage. All entities can contain information about the agents who are responsible for them, the time when they were created and the timespan they are valid, further each entity is connected to the activity which was used to create the entity. All activities can contain information about the agents who are involved in them and the time or timespan when the activity was performed. The generic development process is shown in Fig. 4. The gathered information during the business understanding, data understanding and data preparation stages are the basis for the creation of modeling activities and entities of the reliability assessment approaches, e.g. perturbation options. This connection is represented within the KG with the modelingEntityWasDerivedFrom property, connecting a modeling entity with the business understanding, data understanding and data preparation entities it is based on. Modeling entities can be adjusted in order to better assess the reliability of a feature. An adjusted evaluation entity keeps track of the original entity it is derived from using the wasBasedOn property, for instance, a perturbation option with better suitable parameters for the problem connects to the old perturbation option including the original parameters.

The deployment view of the generic level is shown in Fig. 5. Deployment entities, representing for instance a reliability assessment, are created for each new input case. These deployment entities use modeling and evaluation

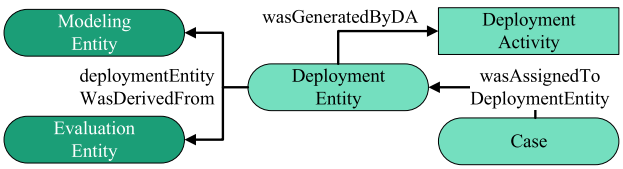

Fig. 5 KG of deployment phase at generic level

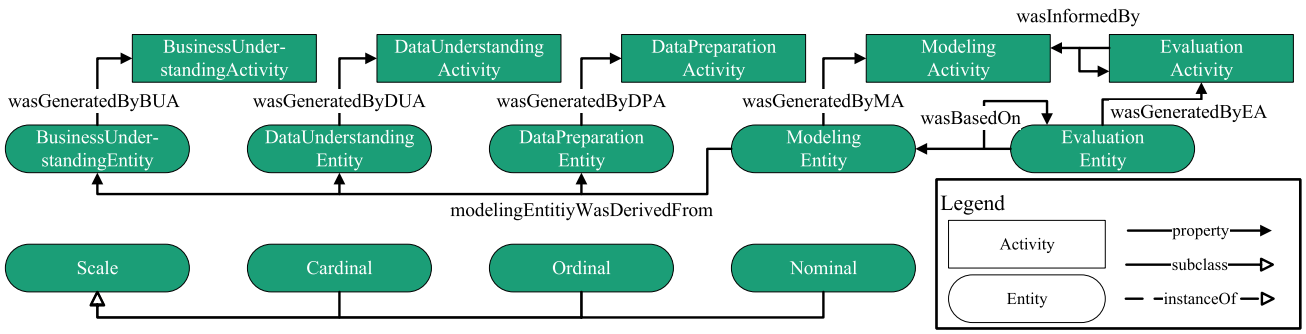

Fig. 4 KG of development phase at generic level

entities, created during development phase, to provide the assessment. This connection is represented with the deploymentEntityWasDerivedFrom property connecting deployment entities with modeling and evaluation entities. Each of these deployment entities may use one or more entities from the modeling or evaluation stage as basis, for instance, two perturbation options to create the perturbed test cases. In this manner, the generic level can be used as basis for method-specific process activities and entities, for example, to create a classification-specific process as shown in Figs. 6 and 7.

In addition to the activities and entities regarding reliability assessment, e.g., a modeling activity or modeling entity, the generic level also includes common knowledge valid for multiple prediction methods. For example, the existence of different scales of feature which is not exclusive only for the classification method. In Fig. 4 three different scales (Scale), Nominal, Ordinal and Cardinal are added as entities to the generic level. If an analyst discovers more common knowledge or common process steps at different method-specific levels, the knowledge can be transferred to the generic level to make it available for future development of new method-specific processes. The adjustment and collection of more common knowledge will be an ongoing process during the creation of future reliability assessment processes and is open for further research to further expand the already collected knowledge. By using a KG instead of, for example, a relational database, we are not bound to rigid data structures and can therefore make adjustments to different processes very easily. This is beneficial for quickly adapting the process to different prediction methods and problem cases as needed.

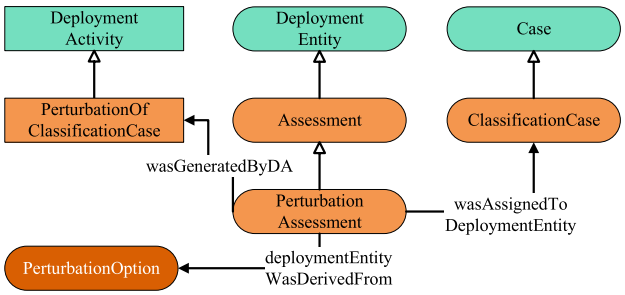

Fig. 7 KG of deployment phase for classification at method level

Method Level

Figure 6 shows an example of the development phase of a reliability assessment process on method-specific level. Activities and entities included in a reliability assessment process on method-specific level for classification, are technically represented as subclasses of the generic activities and entities. Each of these subclasses can have further properties related to them, for instance the time when they were performed or defined, who is responsible for them or how much money is planned for them. The generic information can be further extended depending on the specific entity and the chosen method. For classification of structured data, each subclass of data understanding entity and data preparation entity has a connection to its feature it belongs to. The PerturbationApproach entity can contain basic information about the approach, for example, that an analyst needs access to the training data in order to calculate local measures. The ScaleOfFeature entity captures which of the

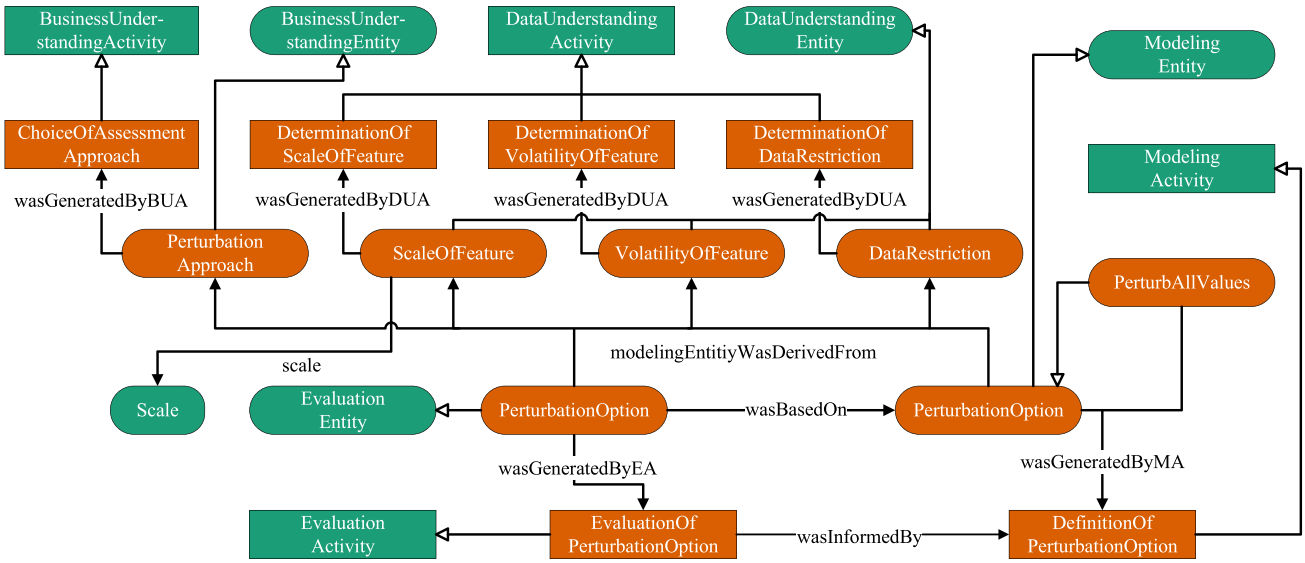

Fig. 6 KG of development phase for classification at method level (color coding according to Fig. 3)

available scales was assigned to a feature. The VolatilityOfFeature entity captures which volatility value was assigned to a feature. The DataRestriction entity can capture a rule or interval indicating the restriction of possible values for a feature. The PerturbationOption entity captures all information needed for its use displayed in the tables in "Perturbation Options", namely the connection to the feature that should be perturbed, the scale of feature which is required, any additional parameters, the perturbation level and the generation algorithm.

Figure 7 displays a reliability assessment process including example subclasses for the deployment part of the method-specific level. An Assessment entity and its subclasses contain the information which is handed over to the analyst to decide if the prediction should be trusted or not, namely the perturbed test cases with their predictions, for example shown in Table 14 or the value for the calculated local measure. Each Assessment entity has a connection to the modeling entities which were used to create the assessment, for example, the perturbation options (PerturbationOption) used to create the perturbed test cases assigned to the perturbation assessment. The ClassificationCase entity captures the input data for the new use case for which the prediction should be assessed, for instance, the data row in CSV format. The method-specific process may be extended if there are further activities or entities important for assessing the reliability of the method's prediction.

It is possible to define subclasses of entities where some of the needed information is already filled in and captured at the method-specific level to be instantiated for different problems. For example, a perturbation option to perturb all available values of a feature is not only useful for the problem it was initially developed, but can be used for all classification problems. A subclass of PerturbationOption is added to the method-specific process where the required scale of feature, the needed parameters and the generation algorithm is already assigned to the specific values for the PertubAllValues option.

Problem Level

The third and last level of the reference process is named problem-specific. Given a method-specific process, it can be used for any prediction problem which uses this method for its predictions. Instances of the method-specific activities and entities can be created for each problem case as shown in Figs. 8 and 9. For example, the method-specific process for classification does contain an activity named ChoiceOfAssessmentApproach, which is now instantiated for our

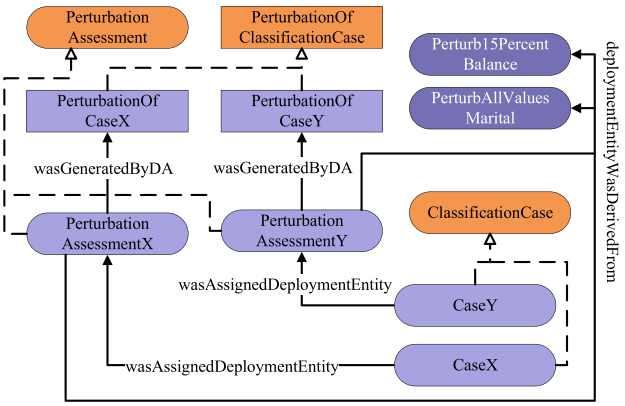

Fig. 9 KG of deployment phase for telemarketing at problem level (Note: Color coding according to Fig. 3)

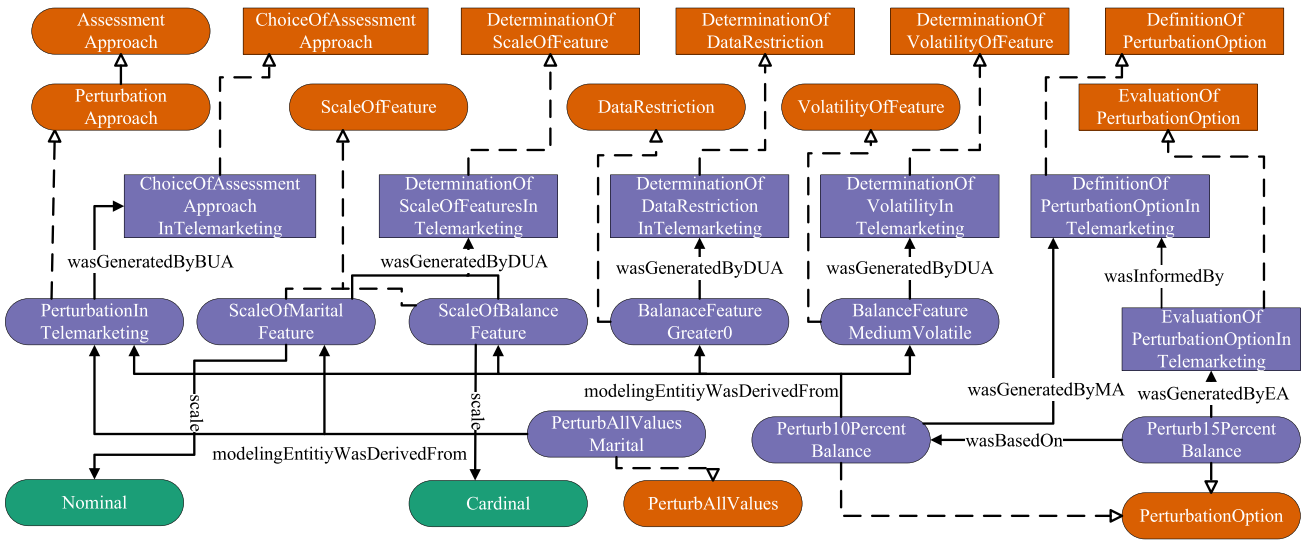

Fig. 8 KG of development phase for telemarketing at problem level (Note: Color coding according to Fig. 3)

telemarketing use case. The project team decides that the *perturbation approach* is used for our telemarketing use case and captures this decision as the PerturbationInTelemarketing entity including all specified properties, for example, the date of the decision and the agents who are responsible for it. All possible data understanding and data preparation steps are instantiated and the gathered knowledge is saved as entities within the problem-specific KG. During the data understanding and the data preparation stage, the scale of the features marital (ScaleOfMaritalFeature) and balance (ScaleOfBalanceFeature) was gathered and captured as Nominal and Cardinal. Assuming that the balance of the used accounts cannot be negative, this data restriction was documented and saved as an entity (BalanceFeatureGreater0). Furthermore, the volatility of the feature balance was stated as medium volatile (BalanceFeatureMediumVolatile) and saved as entity. Given all the information captured in the various instantiated entities, they can be used as indicator for the creation of modeling entities, for example, for the creation of perturbation options shown in "Perturbation Options". Based on these information, the project team can create perturbation options, for example, a perturbation option (Perturb10PercentBalance) to perturb the feature balance in an area $\pm$ 10% around its real-world value based on the information that the feature is cardinal and does have a medium volatility. Figure 16 shows this perturbation option with all assigned properties specified in Table 1.

After the development of the perturbation options, they are used during deployment to help with the assessment of new prediction cases. Figure 9 shows the problem-specific level KG of the deployment phase for telemarketing. Each new ClassificationCase is used together with the perturbation options to receive an assessment about the reliability of the model's prediction saved as entity in the problem-specific KG. Figure 17 shows an example that illustrates the problem-specific KG, including properties for the scenario used for Table 14. Figure 16 shows an example extract from the KG, focusing on the perturbation option to check the 10% radius around a feature.

Business Understanding

Business understanding is the first stage of the CRISP-DM and, therefore, the starting point for a data mining project. The business understanding stage determines the most important project parameters. Besides budget and general project plan the development team defines the project's business goals and data mining goals. In addition to these parameters and goals, the team must decide in which way the reliability of the predictions should be assessed.

The data mining goals influence the decision of prediction method, e.g., classification, which in turn influences the choice of method for reliability assessment. The development team may follow previously defined best practices, including the use of already developed methods for reliability assessment. The development team may, however, also define and document their own method-specific activities, including the design of a reliability assessment method along with capturing the required metadata. For example, the development team may choose among the two predefined reliability assessment approaches *perturbation of input cases* and *use of local quality measures*, which are specific to classification problems. Accordingly, the development team must decide if one or both predefined approaches are suitable in the specific context. In case of the use of local measures the development team would have to ensure access to the training data after deployment in order to be able to calculate the local measures. The *perturbation approach* may be a good fit if the project team is interested in finding sensitive input features. The development team also has to decide if any method-specific activities, including the specific reliability assessment approaches, are suitable for the specific business problem. A predefined reliability assessment approach possibly also requires further adaptions in order to be useful for a particular problem. This paper describes two reliability assessment approaches for the classification-specific process (at the method-specific level of the reference process), namely the use of local quality measures and the perturbation of input cases. Future work will investigate additional approaches and other method-specific processes, e.g., the adaption of the *perturbation approach* for a regression-specific process for reliability assessment.

Example 1 (Choice of assessment approach) In order to be able to apply methods to assess the reliability of individual predictions later the development team first has to choose which method should be employed, if any. The development team may, for example, choose local quality measures for the assessment of reliability, which requires access to the training data of the prediction model. The activities in the subsequent stages must then ensure that the training data are made available for the reliability assessment after deployment. This additional effort—granting access to the training data at all stages—must be considered in the planning stage of the data mining project. Furthermore, the business background may prevent the development team from choosing a specific approach, e.g., due to data privacy regulations, which may prevent the development team to make the training data available after deployment.

The generic level comprises a generic BusinessUnderstandingEntity and BusinessUndetstandingActivity, where the BusinessUnderstandingEntity is generated by a BusinessUndetstandingActivity. On the method-specific level, different subclasses of the generic

BusinessUndetstandingActivity and the generic BusinessUnderstandingEntity are introduced. An example of a method-specific subclass of a BusinessUndetstandingActivity is the ChoiceOfAssessmentApproach. The business understanding task ChoiceOfAssessmentApproach can be represented in the knowledge graph as a specialization of the generic BusinessUndetstandingActivity (Fig. 10). An example of a method-specific subclass of a BusinessUnderstandingEntity is the AssessmentApproach; we describe two potential approaches in this paper, namely *local measure approach* and *perturbation approach*. On the problem-specific level, instances of the method-specific activities and entities are created, which contain concrete values for one specific problem case of the method, for example, the ChoiceOfAssessmentApproachInTelemarketing problem case and the instance of the PerturbationApproach that was chosen for the assessment of the reliability of prediction results in the telemarketing problem case.

Data Understanding

Different data mining projects use different kinds of input data, for example, structured data, streaming data, images, or videos. In order to achieve a certain business or data mining goal, the required data must be collected. During the data understanding stage the required data are collected and the quality of the data must be assessed. While collecting data from various sources, inaccuracies may slip into the values, e.g., due to transmission errors, or it may not be possible to precisely capture a value, e.g., in case of sensor data. Any information about these inaccuracies should be documented in order to be later used for reliability assessment. If the development team cannot quantify the inaccuracies by stating a window which the true value lies within, the development team should at least document that a feature may not represent the precise value. An example for such an inaccuracy of collected data is sensor imprecision [24]. Potential imprecision should be stated for every feature and since there is often more than one method or device involved in the gathering, there can be multiple windows per feature.

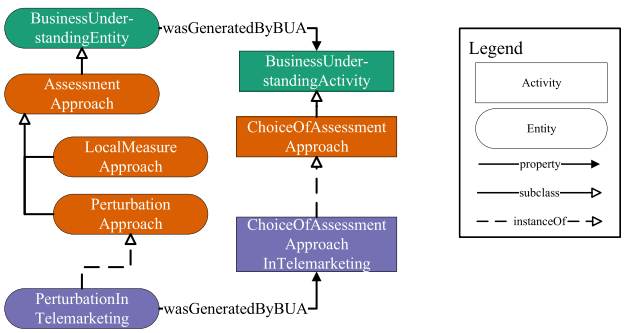

Fig. 10 Example of KG for business understanding

Data understanding and data preparation are necessary steps in every data mining project to come up with a high quality training set for the predictive models. Each data point in the data set has to be extracted from a real-world data source. Furthermore, the data typically have to be cleaned and transformed before they can be used to train a predictive model. In this paper we focus on the use of structured data and leave the investigation of reliability assessment for analyses of other kinds of data to future work. In the following we describe guidelines and possible information sources during the data understanding and data preparation stages, which can then be used to assess the reliability of the predictions later.

Level of measurement Given structured data, each feature can be categorized into one of three levels of measurement (or scale of feature): nominal, ordinal and cardinal. The level of measurement should be determined during the data understanding stage because information regarding the level of measurement can later be used for reliability assessment. Some approaches may be different based on the level of measurement and, therefore, the level of measurement must be recorded.

Example 2 (Level of measurement) It is important to know which level of measurement is assigned to an input feature to apply further reliability assessing actions including calculations. For example, one cannot calculate an area of $\pm$ 5% around a nominal feature such as the marital status. Ordinal features characterize themselves through a given order between the distinct values, for example, the size of t-shirts, where we have small, medium and large which can be clearly ordered from large to small. Cardinal features can be used for any kind of calculations, for instance, an account balance of 1000 $ can be added to a balance of 2000 $.

Feature volatility Different features have a different real-world volatility: some values change more frequently than others. Capturing the real-world value of a feature that is valid for a longer period of time can be difficult when the feature is volatile and its real-world value changes quickly. For example, a sensor measuring the wind speed at an airport may serve a different value compared to the measurement a minute ago. Compared to such volatile features, there are also features that do not change constantly and once their value is determined there is no inherent uncertainty as to the current value. For example, a feature indicating if a customer has already been contacted for an advertisement campaign can assume the values "yes" or "no". There are no values in between and the distinction is clear. If the month of the advertisement phone call is documented, there is no inherent uncertainty about the real-world month.

Example 3 (Feature volatility) Consider the monthly salary of a self-employed worker. Depending on the current work situation, the monthly salary will vary from earning a considerable amount (good work situation) to earning very little or even nothing (bad work situation), which leads to a volatile monthly salary. On the other hand, consider the education of a person in their thirties. The probability that there will be a big change in that person's education is rather low. Therefore, the education level will stay constant once a person reaches a certain age. A value being volatile can be a problem when using that value as the basis for a prediction. For example, a model predicts that a person with a monthly salary of 2000 $ (and other feature values) will not be able to pay back a requested loan. The salary being a volatile feature, if the prediction in that case would change if the monthly salary's real value were slightly higher, e.g., 2500 $, and if such a variation in that case were a realistic possibility, the prediction would not be very reliable.

Restrictions on features Depending on the feature, possible value restrictions may be in place. Restrictions can be either formal or domain-specific. Formal restrictions are problem-independent. For example, a person's age cannot be negative. In addition to formal restrictions, there can be domain-specific restrictions that may vary from problem to problem. For example, to open a bank account, a person must be over 18 years old. This restriction applies to a specific problem and restricts the feature values to start at the age of 18.

Example 4 (Restrictions on features) A feature can be subject to certain restrictions such that features cannot take on values beyond the specification of the feature. A data set on different aircraft types may contain the characteristic maximum fuel tank size. For perturbation, this means that the current fuel level at the start of the aircraft cannot be higher than the maximum fuel tank size and, therefore, should not be perturbed beyond that maximum fuel tank size.

Data accuracy The data used in the analysis may have been collected using various methods. Depending on the collection method different kinds of precision can be stated. Any imprecision within the data may pose problems for the reliability of the predictions.

Example 5 (Data accuracy) A value may be rounded when gathering the data. For example, a salary given in hundreds is most likely not the exact value but in reality slightly less or more due to various reasons. Therefore, it is important to note the possible range of rounded values. Given the salary in hundreds, e.g., 1 000 €, the original interval may be [950 €, 1 050 €] and the real-world salary is anywhere within this interval. If any kind of technical device, e.g., a sensor, is used to collect the data, the accuracy of the device should be stated. This information may be as absolute values, for example, the measured value is correct within the range ± 0.5, as relative value, e.g., ± 5%, or as accurate to a certain number of decimal places.

Figure 11 shows an example of how data understanding activities and entities are represented in the knowledge graph. The generic level represents a generic DataUnderstandingActivity and DataUnderstandingEntity, where the DataUnderstandingEntity is generated by a DataUnderstandingActivity. On the method-specific level, different subclasses of the generic DataUnderstandingActivity and DataUnderstandingEntity are introduced. Examples of method-specific subclasses of a DataUnderstandingEntity are: the ScaleOfFeature, a DataRestriction, or the VolatilityOfFeature. Examples of method-specific subclasses of a DataUnderstandingActivity are: the DeterminationOfScaleOfFeature, the DeterminationOfDataRestriction, or the DeterminationOfVolatilityOfFeature. On the problem-specific level, instances of the method-specific activities and entities are created, which contain concrete values for a specific problem case, for example, the activity DeterminationOfScaleOfFeaturesInTelemarketing, or the entity ScaleOfBalanceFeature containing the scale of the feature *balance*, which is indicated to be Cardinal.

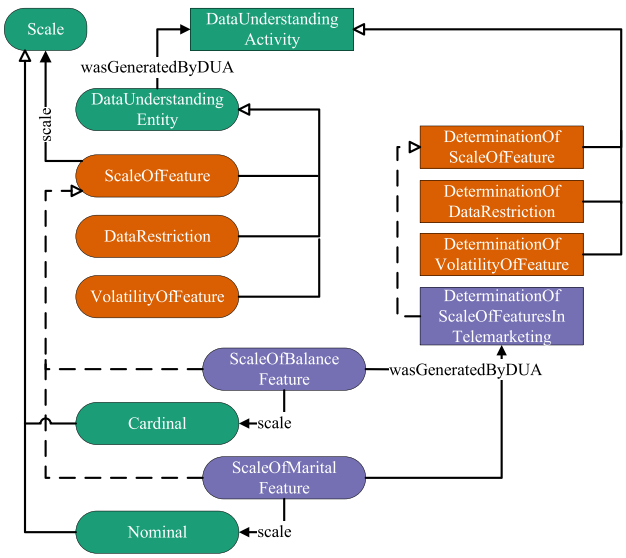

Fig. 11 Example of KG for data understanding

Data Preparation

Once the data collection is finished and all the required data are available, the data preparation stage consists of applying different techniques to improve the data quality and to transform the data into a format suitable to serve as training data for a predictive model. Data quality can be improved along multiple directions: in terms of accuracy, consistency, incompleteness, noise or interpretability. The used techniques may alter or estimate feature values of training samples. The used techniques and the applied transformations should also be documented because they can be a reason why inaccuracies compared to the raw real-world values are included in the final training data set. Examples for such techniques may be to bin a feature into a defined number of groups for numerosity reduction or the imputation of mean values for missing values. The bin value summarizes all data points in the bin and may differ from the real-world value lying anywhere within the range of the bin. Furthermore, the imputation of estimations for missing values may be another potential concern when assessing the reliability of a prediction. In the following, we enumerate several example cases where reliability-related information should be collected during data preparation.

Binning Numerosity reduction can be helpful to better find patterns within a data set. One numerositiy reduction technique is binning. The value space of a feature is divided into areas named bins. Each data point within a bin is substituted by the same value representing this bin, for example, the middle value of the bins' range. Since the binned value is an estimate of the original value and may serve in later steps to assess the reliability of a prediction it should be documented together with the bin range.

Example 6 (Binning) Given that the amount of different values for a feature is very high, it can be fruitful to categorize them into groups. For example, the given monthly salary of possible telemarketing customers can be different up to two decimal places. Therefore, it is a common technique for numerosity reduction to assign each value in a previously defined bin. These bins can be, for example, [0$, 1500$[, [1500$, 5000 $[, and [5000 $, ∞[and for each member of a bin, the mean value of all the members is used. For instance, if the bin [0$, 1500 $[contains the values 500 $, 1000 $ and 1200 $ then all of the three examples will be used with the mean value of 900 $ for all further tasks. This can lead to reliability problems if a prediction is based on the value 900 $ instead of the real-world value which is anywhere within the bin range.

Regression functions If a regression function is used during data preparation to smooth a noisy feature the evaluation metrics of this regression function, e.g., the root mean squared error (RMSE), should be documented by the development team. The RMSE represents the amount that a predicted value deviates on average from the actual observed values. In later stages of the CRISP-DM, it may be interesting to investigate the reliability of the predictions using the collected regression evaluation metrics.

Example 7 (Regression functions) In order to smooth a noisy feature, a regression function can be used to retrieve a feature value instead of the actual value. Assuming that a regression function for the amount of needed fuel of an aircraft is fitted depending on the travel distance, the aircraft type and the payload, the RMSE can be used as good indicator for reliability assessment, for example, used as range in a perturbation option.

Missing values Various techniques exist to handle missing values in a raw data set. One of the most basic techniques is to simply impute the mean value of the feature for any missing value. There are also more sophisticated methods to deal with missing values, e.g., Markov chains. The employed method for handling missing values should be documented and may be later used for reliability checking.

Example 8 (Missing values) The handling of missing values is important in order to train a reliable prediction model. Assuming that during training of the model a specific method to substitute missing values in the training set was chosen, that same method should also be used for handling missing values in new prediction cases. For instance, if the mean value of all monthly salaries is used for missing monthly salary values, then this value should also be used for missing monthly salary values in new prediction cases. Since the imputation of a missing value is just an estimate, the real value can deviate from this estimate which may lead to a changed prediction.

The list of data preparation techniques described in this section is not exhaustive and can be extended in the context of a specific project. In general, all of the performed techniques during data preparation that alter or transform the data set should be documented. This information may serve as input for different reliability assessment approaches in the later stages of the CRISP-DM as described in "Modeling and Evaluation" and "Deployment".

Figure 12 illustrates how data preprocessing activities and entities are represented in the knowledge graph. The generic level provides a generic DataPreparationActivity and a generic DataPreparationEntity, where the DataPreparationEntity is generated by a DataPreparationActivity. On the method-specific level, different subclasses of the generic DataPreparationActivity and DataPreparationEntity are

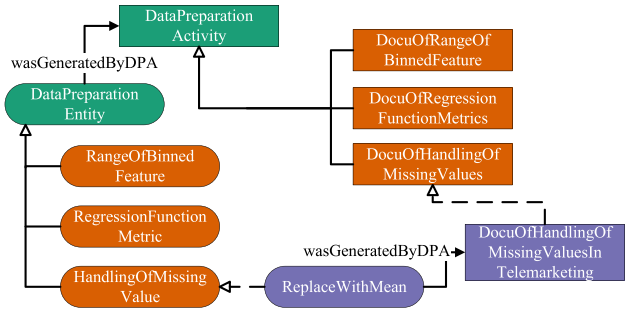

Fig. 12 Example of KG for data preprocessing

introduced. An example of a method-specific subclass of a DataPreparationEntity is the routine for handling of missing values (HandlingOfMissingValue). An example of a method-specific subclass of a DataPreparationActivity is the documentation of how missing values are handled in the data set (DocuOfHandlingOfMissingValues). On the problem-specific level, instances of the method-specific activities and entities are created, which contain concrete values for one specific problem case of the method, for example, the activity to document the handling of missing values in the telemarketing problem case (DocuOfHandlingOfMissingValesInTelemarketing), or the entity containing the approach which was used to handle missing values within the telemarketing problem case, for example, to replace missing values with the mean value of the column (ReplaceWithMean).

Modeling and Evaluation

In this section we describe the development of two reliability assessment approaches, namely the use of perturbed input cases and the use of local measures. Depending on the used data analysis method (clustering, classification, regression, etc.) and the specific prediction problem—for the method classification, for example, this can be prediction of defaulting loans, forecasting whether it will rain on the next day or not, predicting flight delay risk classes, etc.—different approaches using different parameters can be useful. The implemented approaches can then be used by business users during deployment to get an insight into the reliability of a specific prediction case. In this paper, we describe two reliability assessment approaches in detail: the perturbation of test cases and the use of local quality measures.

Perturbation of Input Cases

Our *perturbation approach* consists of two main elements, namely the perturbation options and the perturbation modes. We further describe which and how information gathered in during data understanding and data preparation stages influences the creation of perturbation options.

Perturbation Options

A perturbation option describes an algorithm to generate neighboring values for a given input value. Each perturbation option is assigned a feature from the data set. The value of this feature is the input value for which the neighboring values are created. The thus created neighboring values are then inserted, instead of the original value, into the original case resulting in perturbed test cases. Furthermore, the thus created perturbed test cases are handed over to the model and receive predictions. A user can examine, if the predictions for these cases are different compared to the prediction of the original case. Regarding this approach we took inspiration in the field of numerical mathematics, where the condition number measures the effect of a changed input to the output of a given function [6]. Depending on which algorithm the input value should be altered, additional parameters may be needed, for example, the precision of a sensor used during data collection or the borders of a search interval. Several influencing factors for the creation of perturbation options and their assignment to features are described in "Perturbation Options". Furthermore, each perturbation option is assigned one of three perturbation levels, namely red—perturbed test cases must not change the prediction because one cannot be sure about the real-world value within the perturbation range—orange—perturbed test cases may change the prediction because a borderline case between two classes can be reached—or green—perturbed test cases are expected to change the prediction. These perturbation levels may indicate how critical a changed prediction, based on the perturbed values might be, assisting the analyst during the assessment of the reliability of a prediction.

In the following we represent a collection of possible perturbation options. A perturbation option has the properties name, scale of feature, perturbed feature, additional values and the perturbation level. The scale of feature is important for the generation algorithm because it is not possible to perform calculations with nominal and ordinal data. The perturbation level depends on characteristics of the perturbed feature and is therefore changeable for each option. For example, a perturbation option to perturb an input value by $\pm$ 10% can be used to perturb a highly volatile feature which may change the prediction (perturbation level orange), but also to perturb the precision range of a measured feature which must not change the prediction (perturbation level red). The generation algorithm then takes as input the original feature value of a use case, which is referred to as *origValue* in the algorithm. The algorithm systematically alters a feature value of the input

Table 1 Perturbation option 10%

Perturbation option	
Name	percent10
Scale of feature	cardinal
Perturbed feature	featureName
Additionally required values	–
perturbation level	Orange
Generation algorithm	

for(i = 1; i <= 10; i++){
nextPertVal(origValue * (1 + $\frac{i}{100}$)
nextPertVal(origValue * (1 - $\frac{i}{100}$)
}	

case, repeatedly invoking the *nextPertVal* function to specify another perturbation of a feature value. The presented collection of perturbation options is not exhaustive and may be extended for each new problem.

Table 1 shows a perturbation option for a cardinal feature value. The input value is increased and decreased by 10% in steps of one percentage points. Applying this perturbation option to a feature value will choose the perturbation range relational to the the given value. Lower input values will be perturbed within a smaller range compared to higher input values.

Table 2 shows a perturbation option intended to be used for a feature captured by a sensor. In addition to the feature value, the precision of the sensor in per cent should be handed over the perturbation option. Since one does not know the exact real-world value, it might be good to have a look at more values within the possible range indicated by the precision of the sensor. If the prediction changes for any perturbed test case from the area included in the precision, the prediction for the input case might not be very reliable.

Table 2 Perturbation option sensorPrecision

Perturbation option	
Name	sensorPrecision
Scale of feature	cardinal
Perturbed feature	featureName
Additionally required values	sensorPrecision%
perturbation level	Red
Generation algorithm	

for(i = 1; i <= 10; i++){
nextPertVal(origValue * (1 + ($\frac{sensorPrecision\%}{1000}$ * i))
nextPertVal(origValue * (1 - ($\frac{sensorPrecision\%}{1000}$ * i))
}	

This sets the perturbation level for this perturbation option to red.

Table 3 shows a perturbation option which increases the input value step by step by a specified amount.

Table 4 shows a perturbation option which decreases the input value step by step for a passed amount.

Table 5 shows a perturbation option which takes as input the borders of a desired perturbation range and the number(steps) of perturbed values that should be generated in this range. The range is divided by the steps and a perturbed value for each step is returned.

Table 6 shows a perturbation option for an ordinal input value. The perturbation option requires an ordered array including all possible values for the specified feature and the number of steps how many perturbed values should be created. Depending on the input value, the perturbation option will return perturbed values following the ordering of the feature.

Table 7 shows a perturbation option that can be used with ordinal or nominal features. The perturbation options requires an array including all possible values for the specified feature. It will return all other possible values as perturbed values.

Influencing factors

In order to assess the reliability of a prediction using the *perturbation approach*, an analyst wants to find suitable perturbation options, which tackle possible reliability problems in the data set. We demonstrate how to use the previously gathered information regarding the data set and the applied data preparation steps for the creation of perturbation options. To this end, we discuss the following influencing factors.

Influence of level of measurement While working with structured data, each feature is assigned to one scale of feature. Given these scales, we can apply different perturbation operations on the features. Nominal features do not provide any natural order, which means one cannot take "the next" element or apply calculations on them. One way to perturb a nominal feature would be to swap all available values shown in Table 7. Ordinal features provide a natural order but the distances between the elements can be different. This means that it is also not possible to apply calculations on ordinal features. It is possible to also swap all available values but since there is an order given, one might also use a fixed number of steps in one or both directions, for example, shown in Table 6. Cardinal features can be used in calculations and may be perturbed in relative or absolute manner. Table 2 can, for example, be used together with a cardinal feature to perturb a percentage range around the given value.

Example 9 (Influence of level of measurement) Given that the level of measurement is divided into the three values

Table 3 Perturbation option fixedAmountInSteps+

Perturbation option	
Name	fixedAmountInSteps+
Scale of feature	cardinal
Perturbed feature	featureName
Additionally required values	Amount, steps
perturbation level	Orange
Generation algorithm	

for(i = 1; i <= steps; i++){
nextPertVal(origValue + (i * amount))
}	

Table 4 Perturbation option fixedAmountInSteps-

Perturbation option	
Name	fixedAmountInSteps-
Scale of feature	cardinal
Perturbed feature	featureName
Additionally required values	Amount, steps
perturbation level	Orange
Generation algorithm	

for(i = 1; i <= steps; i++){
nextPertVal(origValue - (i * amount))
}	

Table 5 Perturbation option perturbRange

Perturbation option	
Name	perturbRange
Scale of feature	cardinal
Perturbed feature	featureName
Additionally required values	lowerBound, upperBound, steps
perturbation level	Orange
Generation algorithm	

range = upperBound - lowerBound
for(i = 1; i <= steps; i++){
nextPertVal(lowerBound + ($\frac{range}{steps}$ * i))
}	

nominal, ordinal and cardinal, different perturbation options are useful depending on the actual level of measurement of a feature. Nominal features do not have any kind of ordering, nor cannot be done any calculations with these values. Therefore, the number of perturbation options are limited to algorithms not depending on these properties. For instance, it is always possible to perturb a feature value by all available values. The feature marital status contains the values single, married and divorced. Any given value can be perturbed by the other two. This is also be valid for ordinal features. They are also offering the possibility to chose the next value instead of a random one. For example, consider the size of a t-shirt. Given that the input case has a size value of *small*, it may be more useful to perturb the next value in the order before using a random value from the value range of the feature. A perturbation utilizing the order of the feature can be used, to receive the neighboring values which would be *medium* in this case. Considering cardinal values the use of a perturbation option inserting all available values is not optimal, thinking of the huge range of different values for any kind of number like the monthly salary or the account balance. Therefore, other kinds of perturbation options can be used to receive the best opportunity of assessing the reliability of a prediction. For instance, use a fixed value or a percentage value as perturbation range, where possible perturbed values are chosen from. Assuming that the monthly salary is always rounded to hundreds, it is advisable to check more values around the feature value given for the prediction. The real-world value of a monthly salary of 1200 $ may be anywhere between 1150 $ and 1250 $. Choosing five values below 1200$ and five values above raises an analyst's confidence that the prediction is not changing any of the possible real-world salary values.

Influence of feature volatility The real-world volatility of a feature may be a good indicator for the creation of a perturbation option. If a value is known to be very volatile in the real world, it might be helpful to perturb this value within a suitable range based on the approximate range of the volatility using the perturbation option shown in Table 5. Given a very stable feature, it might be less likely that there are inaccuracies in the value and a smaller range of a perturbation option used together with this feature may be suitable.

Example 10 (Influence of feature volatility) Given a feature value where the real-world value is not known within an interval or cannot be determined properly is a good indicator for a perturbation option. This is the case for highly volatile features. Example 3 mentioned the monthly salary as an example for a potentially highly volatile feature. Following other classification examples, for example, in [25] where the goal was to predict a delay class for flight scenarios, the wind speed measured at a specific time is considered as highly volatile. Measuring the wind speed a few times multiple seconds apart, can yield highly different values. Therefore, it is suggested to perturb the received value within a chosen interval. The interval is chosen using domain knowledge about the perturbed feature, for example, assuming that a measured wind speed of 85 km/h may vary about 20 km/h within one minute, a perturbation option selecting multiple

Table 6 Perturbation option perturbInOrder

Perturbation option	
Name	perturbInOrder
Scale of feature	ordinal
Perturbed feature	featureName
Additionally required values	steps, values[]
perturbation level	Orange
Generation algorithm	

```
index = values.findIndexOf(origValue)
size = values.getSize()
for(i = 1; i <= steps; i++){
if(index - i > 0)
 nextPertVal(values[index - i])
if(index + i < size + 1)
 nextPertVal(values[index + i]))
}
```

Table 7 Perturbation option perturbAllValues

Perturbation option	
Name	perturbAllValues
Scale of feature	nominal & ordinal
Perturbed feature	featureName
Additionally required values	values[]
perturbation level	Orange
Generation algorithm	

```
index = values.findIndexOf(origValue)
size = values.getSize()
for(i = 1; i <= size; i++){
if(i != index)
 nextPertVal(values[i])
}
```

values from the interval [65 km/h, 105 km/h] is highly valuable to asses the reliability of a prediction using the input case with the wind speed of 85 km/h. This example shows that information can also be dependent on the time. Following the weather forecast, one can define different volatility levels for different time periods. For instance, the information that wind gusts are less likely in the afternoon can change the volatility level from highly volatile to medium volatile and therefore changes the perturbation range for this feature during the assigned time period.

Influence of restrictions on features Formal or domain-specific value restrictions might be in place for features included in the data set. If a perturbation option would return a value which is violating any given restrictions, the perturbed test case can be discarded. This can reduce the number of total perturbed test cases and fasten the *perturbation approach*.

Example 11 (Influence of restrictions on features) Perturbation options are following a defined algorithm to perturb a given value in a defined manner. This generic problem-independent algorithm is not aware of possible data restrictions, which can restrict the amount or interval of possible values for a feature. For instance, we want to use a perturbation option on the feature age. Given the use case that we want to sell a long-term deposit to customers, we have to consider possible value restrictions about the age feature. If a perturbation option would result in an age which is not possible, for example, −1 years old, there is no need to retrieve an prediction for this perturbed test case. The same applies if there may be legal restrictions in place, for instance, that a person has to be over 18 years old to be allowed to make a long-term deposit. Since the prediction of each perturbed test case is using computing time, we recommend to restrict the prediction of perturbed test case only to valid examples. From the last example follows that perturbed test cases, including the age values 17, 16 and 15 would be discarded and not forwarded to the model to receive a prediction. This can speed up the whole application of perturbed test cases for a specific input case.

Influence of data accuracy The accuracy of a feature may be a good starting point for the definition and assignment of perturbation options. If the accuracy values of the features are documented, it might be a good idea to perturb the given values within the interval of the accuracy. In case the we do not know the exact real-world value the final prediction should not change given any of the values within the accuracy interval. Perturbation options dealing with this kind of information should be assigned a red level. The creation of an accuracy-related perturbation option for any kind of known imprecision included in feature values, e.g, the perturbation options in Tables 1 and 5, is useful for reliability assessment.

Example 12 (Influence of data accuracy) If it is possible to find out the accuracy of a feature value, this is a very good indicator for a perturbation option. For example, if the temperature of a room is measured using a temperature sensor, the given accuracy must be used within a perturbation option to ensure that the prediction does not change within the accuracy interval of the sensor. A value of 20 °C measured with a precision of 10% below or above must be perturbed within the interval [18, 22] °C. If the final prediction changes while just chaining values within this interval, the final prediction is not reliable since one cannot be sure

about the exact real-world value of the temperature due to the precision of the sensor.

Influence of binning During data preparation binning can be applied to continuous features for numerositiy reduction. Applying this technique replaces the real-world value of a feature with a value representing the whole bin, for instance the mean value. This concludes that the exact real-world value is not known anymore and lays anywhere within the range of its assigned bin. A perturbation option, for instance, shown in Table 5, can be used to perturb the range of the binned value to ensure that choosing any other possible value from the bin does not change the prediction.

Example 13 (Influence of binning) Given a data set where the monthly salary is binned in multiple different bins with varying intervals. A new input case contains a salary value of 900$ representing the mean value of all elements with the bin ranging from [0$,1500$[. A perturbation option assigned the the monthly salary should create multiple perturbed values within the bin range. The prediction must not change using any of the perturbed values since one cannot be sure about the original real-world value within the range.

Influence of regression function metric To avoid storing a large number of different values, a regression function can be trained instead of these values, which then predicts the values as needed. This can pose a problem for the reliability of predictions based on this value, since the original real-world value is no longer used but the one estimated by the regression function. A perturbation option can be used to examine the area around this predicted value, for instance, using the perturbation option shown in Table 5 with the the RMSE as lower and upper bound. Metrics of the regression function can be used as an indicator of the radius of perturbation.

Example 14 (Influence of regression function metric) A data set contains multiple features about flights, including the fuel required for the flight. Assuming that a regression function is used to predict the required fuel, based on the features travel distance, aircraft type, and payload, instead of storing each individual fuel value, one cannot be sure about the exact real-world fuel value anymore. A perturbation option can be used to the create perturbed test cases using similar fuel values. The RMSE serves as a good indicator for the range of the perturbation.

Influence of missing values The handling of missing values is necessary in the majority of existing data sets. Multiple approaches, for example, using the mean value of a feature are available to replace missing values with the most probable value. Since one cannot be sure about the real-world value when using a replaced value, this can pose a problem for the reliability of predictions based on this value. The use of a perturbation option that produces perturbed values is recommended to get an assessment of the reliability of this value, for instance, using the perturbation options in Table 3 and 4 to perturb the value upwards and downwards.

Example 15 (Influence of missing values) Given a data set of weather data that is to be used to predict whether it will rain the following day. The input data of today's weather has a missing value for the temperature and this is replaced with the mean value of the last 30 days. Since one cannot be sure whether this mean value corresponds to the real-world value, a perturbation option can be created to receive multiple values similar to the given mean value. If changing this mean value leads to a changed prediction in the end, the reliability of this prediction has to be further investigated.

Perturbation Modes

The more perturbed test cases are examined the better the reliability can be assessed in the end. In the ideal case, each potential possible value of a feature should be used to create perturbed test cases and to receive a prediction for them. Furthermore, only perturbing the cases feature by feature excludes possible combinations of different perturbed features. Therefore, it is also necessary to create perturbed test cases where two or more features are replaced by a perturbed value. Since it is not practical and sometimes not even possible to create all potential perturbed cases due to restrictions regarding execution time or number of cases (number of perturbed test cases grows exponentially with each additional perturbed feature value), it is necessary to specify an operation mode. The operation mode regulates the order in which perturbed test cases are created and predicted by the model. Different prediction problems might need different perturbation modes depending on the urgency of the reliability assessment. Each perturbation mode creates perturbed cases until either all possible combinations of different perturbed values,created by a perturbation option, are provided, or a specified maximum execution time is reached.

We propose three different perturbation modes, namely *full*, *prioritized* and *selected*. To demonstrate the order in which the perturbed cases are created, Tables 8, 9, and 10 show examples of perturbed values resulting from three perturbation options applied on the features age, marital and default. The first line represents the original value and the following rows represent the perturbed values resulting from the perturbation options used on the original feature from the first line.

Full mode The first operation mode is *full*, which computes all possible combinations of perturbed values from the used perturbation options. The mode starts by iterating

through all perturbed values received from the selected perturbation options. The order in which the perturbed values are used for the creation of the perturbed cases is not defined specifically. Each perturbed value of each perturbation option is inserted into the original case for the specified feature. The thus created perturbed cases are then forwarded to the model to receive a prediction. Once more than one feature of the data set should be perturbed, all combinations of perturbed values are created. The full mode stops if all combinations are created or a maximum execution time is reached.

Example 16 (Full mode) Table 11 shows an example of the creation of perturbed test cases. The first row represents the original case, whose prediction should be analyzed regarding reliability. Each following row represents a perturbed test case. The mode starts with the perturbed values for age and iterates through all values. When all perturbed values of age are used, the next feature will be replaced with its perturbed values. Furthermore, all combinations of perturbed values will be created.

Prioritized mode The second operation mode we introduce is named *prioritized*. Keeping in mind that the number of perturbed cases can grow exponentially, the prioritized mode gives the user the option to prioritize features. This means that perturbed cases including a perturbed value of a respective case's features will be created in advance.

Example 17 (Prioritized mode) For example, if an analyst knows that the feature age is important during the analysis, it can be stated that all perturbed test cases including perturbed values of age should be created before the remainder of the cases. Table 12 shows example cases. Since age was flagged as *prioritized* all perturbed cases, including a perturbed value for age, are created in advance. In comparison to the *full* mode, the resulting perturbed test cases are identical but with a changed order, given that the creation was not stopped by maximum execution time.

Selected mode The *selected* mode allows the analyst to select a number of features which should be used for the creation of perturbed test cases. Instead of generating all possible perturbed cases, only combinations of the selected features are used resulting in a subset of cases compared to *full* or *prioritized* mode. Specific features or specific feature values that are not interesting from a business perspective can be skipped for the benefit of faster execution time or to receive an assessment using only a defined number of features.

Example 18 (Selected mode) Table 13 displays an example where only age and default are selected for perturbation. The perturbation option for default is still available but assumed to be of minor importance for this exact use case. All combinations of the selected features are created and then used for the final assessment of the prediction.

After the first definition of the perturbation options based on the gathered information these perturbation options need to be evaluated if they fit for the given prediction problem. Parameters included in the perturbation options may need further adjustments depending on the actual prediction problem they are intended to. During the evaluation stage of the CRISP-DM analysts should assess if given perturbation ranges are too narrow or broaden. For example, changing the percent values/absolute values when they are not suitable for the given prediction problem. It should also be assessed if further perturbation options may be helpful, even though they have not been created yet and must still be developed by the development team.

Figure 13 illustrates how modeling and evaluation activities as well as entities are represented in the knowledge graph. The generic level comprises a generic ModelingActivity, a generic EvaluationActivity, a generic ModelingEntity, and a generic EvaluationEntity, with the ModelingEntity being generated by a ModelingActivity and the EvaluationEntity being generated by an EvaluationActivity. In order to be able to evaluate an entity, this entity has to be modeled in the first place, therefore, an EvaluationActivity is informed by the ModelingActivity, or EvaluationActivity which generated the entity currently under evaluation. A new EvaluationEntity is connected to the ModelingEntity or EvaluationEntity which was the starting point for the current evaluation. On the method-specific level, different subclasses of the generic modeling and evaluation activities and entities are introduced. Examples of method-specific subclasses of a ModelingEntity or EvaluationEntity are the definition of perturbation options (PerturbationOption) as shown in Tables 1, 2, 3, 4, 5, 6 and 7. A PerturbationOption (modeling or evaluation entity) can be used to derive further subclasses already containing some specific values, for example, a PerturbAllValues perturbation option where the generation algorithm is already defined. An example of a method-specific subclass of a ModelingActivity is the DefinitionOfPerturbationOption. An example of a method-specific subclass of an EvaluationActivity is the evaluation whether a perturbation option is appropriate for a given use case or not (EvaluationOfPerturbationOption) and, therefore, may need parameter changes. On the problem-specific level, instances of the method-specific activities and entities are created, which contain concrete values for one specific problem case of the method, for example, the activity to define a specific perturbation option in the telemarketing problem case (DefinitionOfPerturbationOptionInTelemarketing), or the entity Perturb10PercentBalance which

Table 8 Resulting values from perturbation option for feature age

Values for feature age
20
21
19
22

Table 9 Resulting values from perturbation option for feature marital

Values for feature marital
single
married
divorced

Table 10 Resulting values from perturbation option for feature default

Values for feature default
no
yes

Table 11 Operation mode: full

age	job	marital	default
20	technician	single	no
21	technician	single	no
19	technician	single	no
22	technician	single	no
20	technician	married	no
20	technician	divorced	no
20	technician	single	yes
21	technician	married	no
21	technician	divorced	no
...	...	...	...

Table 12 Operation mode: prioritized

age	job	marital	default
20	technician	single	no
21	technician	single	no
19	technician	single	no
22	technician	single	no
21	technician	married	no
19	technician	married	no
22	technician	married	no
21	technician	married	no
19	technician	divorced	no
...	...	...	...

contains concrete parameter values related to the telemarketing problem case. An example of a problem-specific EvaluationActivity is the evaluation of an existing perturbation option in the telemarketing problem case (EvaluationOfPerturbationOptionInTelemarketing) which generates a new perturbation option that has adjusted the parameters of the perturbation option under evaluation, for example, to adjust the perturbed area above and below a given value to 15% instead of 10%.

Use of Local Quality Measures

The second reliability assessment approach which we address in this paper is the use of local quality measures. The *local measure approach* can be based on a comparison and needs access to the training data set because the data is needed in order to calculate the measures. In this approach, the analyst compares global model measures with measures computed only on a subset of data around a new input case. An example of a local quality measure is the local accuracy [26]. The global accuracy indicates the percentage of how much input cases did get predicted correctly during the evaluation stage of the model overall whereas the local accuracy indicates the percentage of correct predicted input cases in a specific subspace of the input data. It may be the case that the quality of the model is different in different subspaces of the input data. Consider a global accuracy of 85% for a given model. Some areas of the input space might have a better accuracy (> 85%) and some might have a worse accuracy (< 85%) In order to calculate a local measure, one needs to define the area that should be considered for the calculation. This can either be done by defining a fixed number of data points around the new input case or by defining a range where all data points within this range are considered. The range can either be a global distance for all given features, for instance, the Euclidean distance or given individually for each feature, for instance, ± 10% for a salary feature and ± 5 years for an age feature. In this way one can get an insight into how well the model performs in the input data space around the new input data case compared to the overall accuracy of the whole model. If the local accuracy is much lower than the indicated overall accuracy an analyst should have a closer look on the reliability of this prediction. For example, a new input case in a feature space with only 60% local accuracy compared to 85% for the whole model has a significantly worse performance than stated by the global measure, which might be a requirement for the model to be used in this business use case. While using local measures, it must be assured that a reasonable amount of cases is considered for the calculation. This might be different from case to case depending on the sparseness of the area around the examined use case. If the number of affected cases is

Table 13 Example perturbation in *selected* mode

age	job	marital	default
20	technician	single	no
21	technician	single	no
19	technician	single	no
22	technician	single	no
20	technician	single	yes
21	technician	single	yes
19	technician	single	yes
22	technician	single	yes

too low, the radius of the calculation should be increased or a different reliability assessment approach should be used instead of local quality measures to assess the reliability of this prediction.

Example 19 (Local accuracy) The global accuracy is one of the most often used metrics indicating the quality of a classification model. It is common practice to use different data samples for the evaluation of prediction models than the ones which are used for training. The samples used for the evaluation are the ones taken into account when calculating the global accuracy of the prediction model. For example, in Fig. 14, all points (*green/striped* and *blue/dotted*), representing test samples, are used to calculate the global accuracy of the model, assuming to be 85%. If there is a new prediction case for which the local accuracy should be calculated, only test samples in a defined range around the new prediction case are considered to calculate the local accuracy of the new prediction case. Assuming that the local accuracy is 60% — only considering the *blue/dotted* points within the range — the reliability of the prediction for this new prediction case is significantly worse than indicated by the global accuracy of 85% for the overall model. The prediction should be marked and an analyst should assess if this worse performance represents a problem for the final prediction decision.

Another example for a local measure is the local class ratio. Since accuracy does not differentiate between different classes it may be helpful to get an insight into the distribution of the classes in the local area. We are interested in how many of the neighboring training cases have the same label as our prediction. Therefore, we need to specify the range which is used for the calculation of the ratio. This can be given as absolute number of training cases that should be used or a distance around the prediction case. If the number of affected cases is too low, the radius of the calculation should be increased or a different reliability assessment approach should be used instead of the local measure approach to assess the reliability of this prediction.

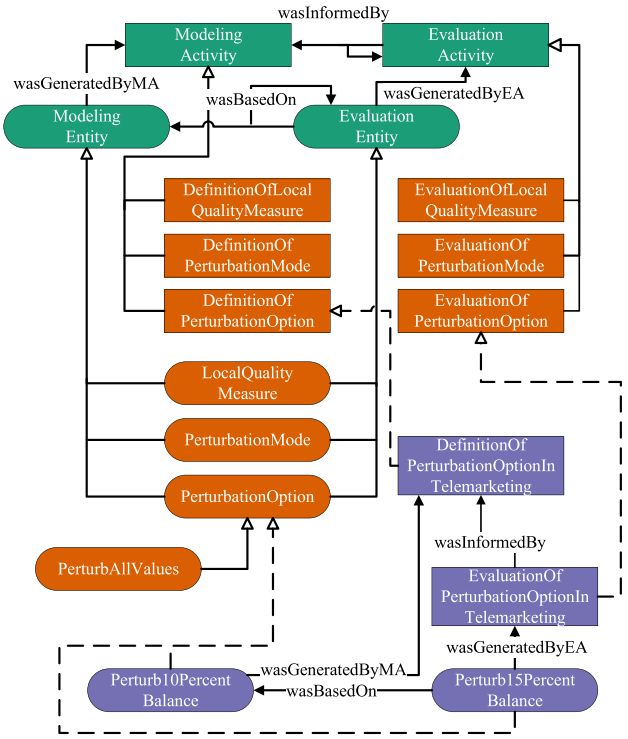

Fig. 13 Example of KG for modeling and evaluation

Example 20 (Local class ratio) The local class ratio is based on the distribution of the labels within the training data. For example, consider Fig. 14 represents the distribution of class labels within the training data. Receiving the prediction of the *blue/dotted* class for the new prediction case, would not lead to doubt about the reliability of the predicted case. If the prediction model predicts *green/striped* for the new prediction case, this case should be highlighted since all training examples in the defined radius around the new prediction case were of a different class than the predicted class.

The examples for local measures given in this paper are not exhaustive. There might be more local measures providing more local insights about the prediction cases depending on the prediction method or the actual prediction problem. If a decision tree is used to classify a case, the distribution of the training labels can be stated for each node of the tree, for example. The next section shows how the proposed approaches are used to assess the reliability for a specific prediction.

Deployment

Deployment is the final stage of CRISP-DM. The trained and evaluated model is put into production into the daily

business of the company. New input cases are handed over to the model and will receive a prediction. Analysts will use this predictions to decide on business actions. The previously developed reliability assessment approaches can assist to assess the reliability of these predictions. Depending on the developed perturbation options and the developed perturbation modes, perturbed test cases can be created and the predictions for this perturbed test cases can be analyzed, whether there are any problems while altering the input values of the new prediction case. Furthermore, given the training data, local measures can be calculated and also assist with the assessment how reliable a prediction for the new case might be.

Reliability Assessment

In order the demonstrate the use of the reliability assessment approaches we trained a logistic regression function over the data from a real-world telemarketing campaign [5]. The use case presented here is about predicting if a new customer will make a long-term deposit when contacted by phone. Included in the data set are different features describing the contacted person, e.g., age, marital status, education, contact information, and existence of any kind of loan, but also features regarding previous campaigns. The labels for the prediction are either *yes*, the customer will make a long-term deposit or *no*, the customer will not make a long-term deposit. The feature *previous call duration* was excluded from the model because we do not know the duration of the phone call in advance of calling a person and therefore we cannot use it for the prediction. Three perturbation options were created to perturb the features marital, default and balance and we used the *full* perturbation mode for the creation of the perturbed test cases. Table 14 shows an example of the created perturbed test cases.

Example 21 (Perturbation of input cases) Table 14 shows a new case where it should be predicted whether the person will make a long-term deposit. The first line shows the original values for the person with the model's prediction *no*. The following lines are perturbed test cases, created by a perturbation option. The nominal features *marital* and *default* were perturbed using the perturbation option defined in Table 7 whereas the *balance* feature was perturbed using the perturbation option defined in Table 2, resulting in 125 perturbed test cases, all created with perturbation level orange. 84 of the 125 perturbed test cases returned the same prediction as for the original prediction case and the remaining 41 returned a changed prediction and require further examination. The first six perturbed test cases were generated by the perturbation option used on the *balance* feature. We systematically added and subtracted a percentage value to the balance and the prediction for these cases does not change, therefore this does not pose a problem for reliability. The next two lines shown in the table were generated by perturbing the *marital* feature. Since the original value was single, the other two available values married and divorced were inserted into the original case, which does not affect the prediction and, therefore does not pose a problem for the reliability of the prediction. The last three shown perturbed test cases include perturbed values for the feature *default*, indicating if the customer has a current credit in default, which is changed from no to yes. As shown in the table, the final prediction changes from *no*, the customer will not make the long-term deposit, to *yes*, the customer will make the long-term deposit. This further means that if our new customer, for whom the prediction of the model was no, had any credit in default, the prediction would change. Considering that the customer may have not given honest information about his other defaulted loans and the fact that our organization may not have access to further information to validate the data, means that the analyst should probably not blindly trust the model's prediction. Despite the prediction that the customer may not make a long-term deposit, an analyst may still consider to contact the customer because of the changed prediction for the perturbed test cases. Following the *full* perturbation mode, all combinations of perturbed values were created and included as perturbed cases in the final report assisting with the assessment of the reliability. All perturbed test cases were created using an orange-level perturbation option. Applying these perturbation options

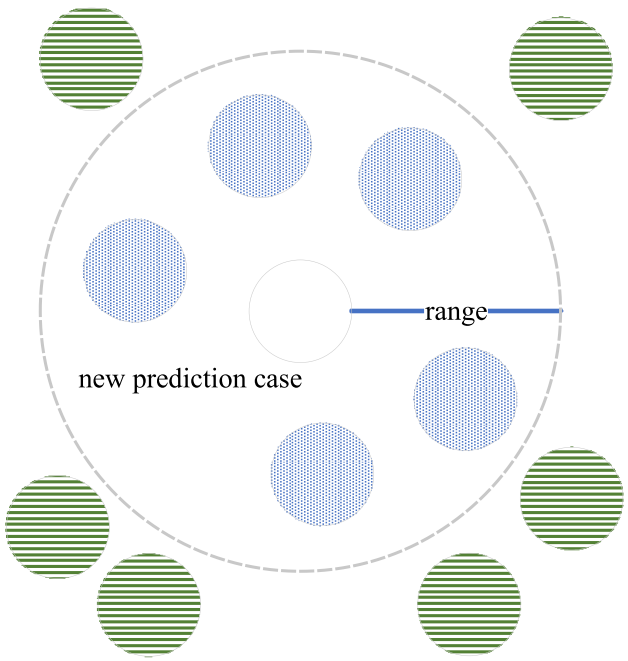

Fig. 14 Calculation of a local quality measure

Table 14 Perturbed test cases for the predictive model over the telemarketing data set

age	job	marital	education	default	balance	housing	...	prediction
20	technician	single	secondary	no	2143.00	yes	...	no
20	technician	single	secondary	no	2164.43	yes	...	no
20	technician	single	secondary	no	2121.57	yes	...	no
20	technician	single	secondary	no	2185.86	yes	...	no
20	technician	single	secondary	no	2100.14	yes	...	no
20	technician	single	secondary	no	2207.29	yes	...	no
20	technician	single	secondary	no	2078.71	yes	...	no
...	...	...	...	...	...	...	...	...
20	technician	married	secondary	no	2143.00	yes	...	no
20	technician	divorced	secondary	no	2143.00	yes	...	no
...	...	...	...	...	...	...	...	...
20	technician	single	secondary	yes	2143.00	yes	...	yes
20	technician	single	secondary	yes	2164.43	yes	...	yes
20	technician	single	secondary	yes	2121.57	yes	...	yes
...	...	...	...	...	...	...	...	...

may lead to a changed prediction due to reaching a label border case in the feature space. In comparison, a red-level perturbation option should not change the prediction in any case, for example, a perturbed sensor value perturbed within the given precision interval.

Figure 15 shows how data deployment activities and entities are represented in the knowledge graph. The generic level comprises a generic DeploymentActivity and a generic DeploymentEntity, whereby the DeploymentEntity is generated by a DeploymentActivity. In addition to the generic DeploymentActivity and DeploymentEntity, there is a generic Case entity representing the new prediction case for which the reliability should be assessed and which is assigned to a DeploymentEntity. On the method-specific level, different subclasses of the generic DeploymentActivity, the generic DeploymentEntity, and the Case are introduced. An example of a method-specific subclass of a DeploymentEntity is the reliability Assessment which is further specified depending on the chosen reliability assessment approach, for example, a LocalQualityMeasureAssessment or a PerturbationAssessment. An example of a method-specific subclass of a DeploymentActivity is the PerturbationOfClassificationCase, or the MeasurementOfLocalQualityMeasures of a new classification case that should be predicted by the classification model. The generic Case entity is used as superclass for different prediction methods, for example, a ClassificationCase. On the problem-specific level, instances of the method-specific activities and entities are created, which contain concrete values for one specific problem case of the method, for example, the activity to perturb a specific problem case X (PerturbationOfCaseX), or the entity containing the perturbation result for the case X (PerturbationAssessmentX). Instances of the method-specific entity ClassificationCase represent the input values of a new prediction case, which are further used within the assessment of the reliability.

We further calculated the local accuracy for the new prediction case to get a better assessment of the prediction for the case.

Example 22 (Local accuracy of test cases) In addition to the *perturbation approach* the *local measure approach* was applied to the use case. The local accuracy for the case shown in the first line of the Table 14 was calculated in order to assess the prediction. The global accuracy for the model is about 78%. Considered the 1500 nearest training set neighbors, measured with the Euclidean distance, the local accuracy for the use case has a value of 93%. This concludes that the model has a better performance in the subspace around the prediction case, compared to the overall performance of the model. Since the overall accuracy of 78% is considered as sufficient, otherwise the whole model would not be used, there is little doubt about the reliability from the point of view of this approach. Having subspaces with a better performance than the overall performance also means that there need to be subspaces with a worse performance, which may be a problem for the reliability of other cases.

Using the Knowledge Graph

The proposed knowledge graph has three main purposes. The first purpose is to allow for recommendations about

which entities and activities should be performed and used together. The second purpose is to support quality management for reliability assessment processes. Finally, the third purpose is to ensure an auditable reliability assessment process.

Recommendation

The designed reliability assessment processes provide the possibility to receive recommendations about the usage of the included activities and entities. For example, a tool built upon the reference process can query information directly from the KG using SPARQL. In this manner, recommendations about possible available perturbation options, based on already gathered information about features can be suggested to a user on the problem-specific level.

Example 23 (Recommendation) A perturbation option that perturbs a measured wind speed value by ± 30 km/h was developed. The feature wind speed has a cardinal scale and is very volatile. If a user is searching for a suitable perturbation option for a feature with the same conditions, a tool can recommend the perturbation option used together with the wind speed feature. Further it does not make sense to recommend or use perturbation options used on features with completely different conditions. For instance, a perturbation option that perturbs all available values of a feature can be used on a nominal feature with a low number of possible values, e.g., the marital status, but it will not be possible to receive a prediction for every possible wind speed value. A tool for applying the *perturbation approach* should not allow the user to use a perturbation option perturbing all possible values in combination with the feature wind speed.

Quality Management

The KG enables a project team to support quality management actions for reliability assessment processes. The knowledge that a specific information about a feature can lead to reliability problems can be used in order to assess the quality of reliability assessment processes. If important information is not used during the assessment of the reliability of a prediction, the assessment approach can be of poor quality.

Example 24 (Quality management) A reliability assessment process using the *perturbation approach* was used on a classification problem of different risk delay classes for flight scenarios. An analyst would like to predict if a possible combination of arriving flights at an airport will have many delays and, therefore, will be costly. Reviewing which features are used during the perturbation, one finds out that the feature *wind speed* was not used in any perturbation option. Knowing that the wind speed feature is very volatile, it should also be perturbed in order to detect possible reliability problems due to an easily changed wind speed. Using an assessment approach that does not address all known possible reliability issues may miss some of them and, therefore, not indicating that the prediction is unreliable.

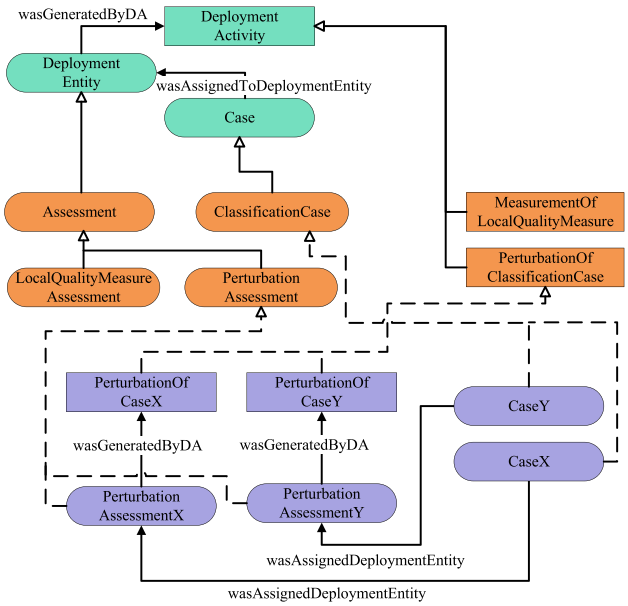

Fig. 15 Knowledge graph example for deployment

Decision Audit

Accurate and precise documented processes ensure an auditable performed reliability assessments. This is especially beneficial if responsibility and traceability are important properties for the prediction problem. All activities and entities, to ensure the reliability assessment are documented within the KG and can be retrieved and displayed if necessary. Companies and external auditors have the possibility to check whether the defined assessment process for a prediction problem has been performed properly for a particular case. It can be audited that decision-makers who base decisions on predictions, followed the assessment process and did not solely rely on the given prediction without performing any additional reliability assessment activities. In addition, it should be examined what consequences have been drawn if the prediction is found to be unreliable.

Example 25 (Decision audit) Consider the example in Figs. 16 and 17. The bank would like to know if the reliability of the prediction for the case has been assessed before a decision was made. By querying the knowledge graph an auditor may retrieve information about reliability

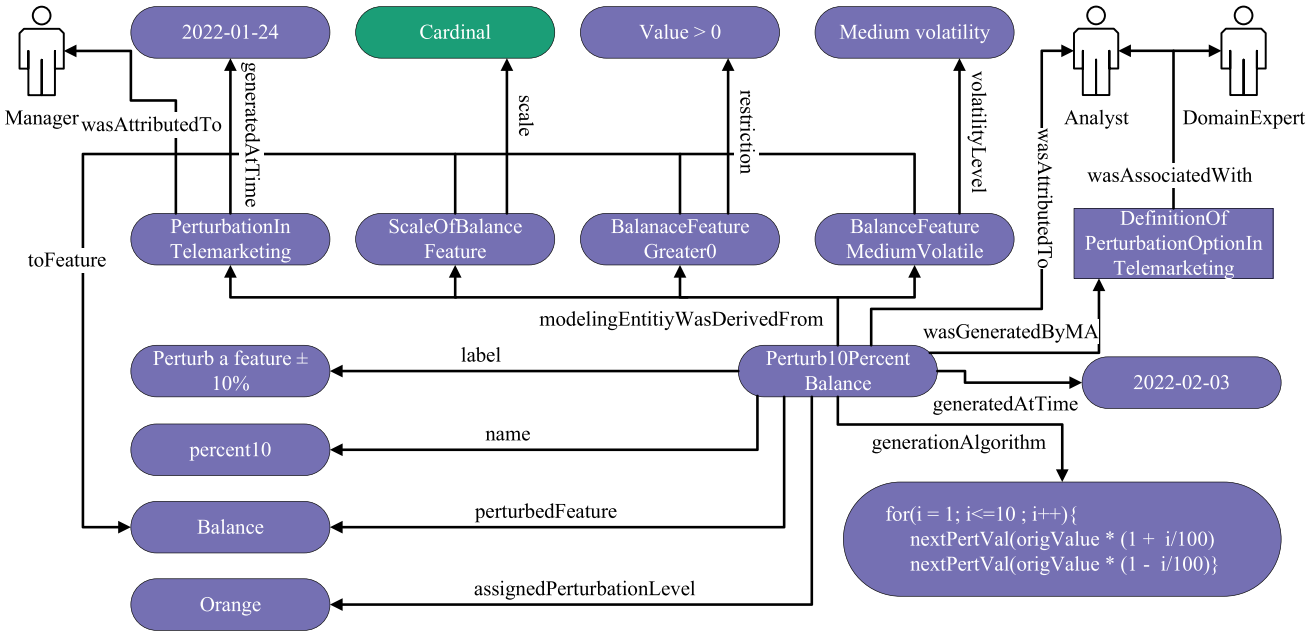

Fig. 16 Example extract of problem-specific level KG of development phase for telemarketing

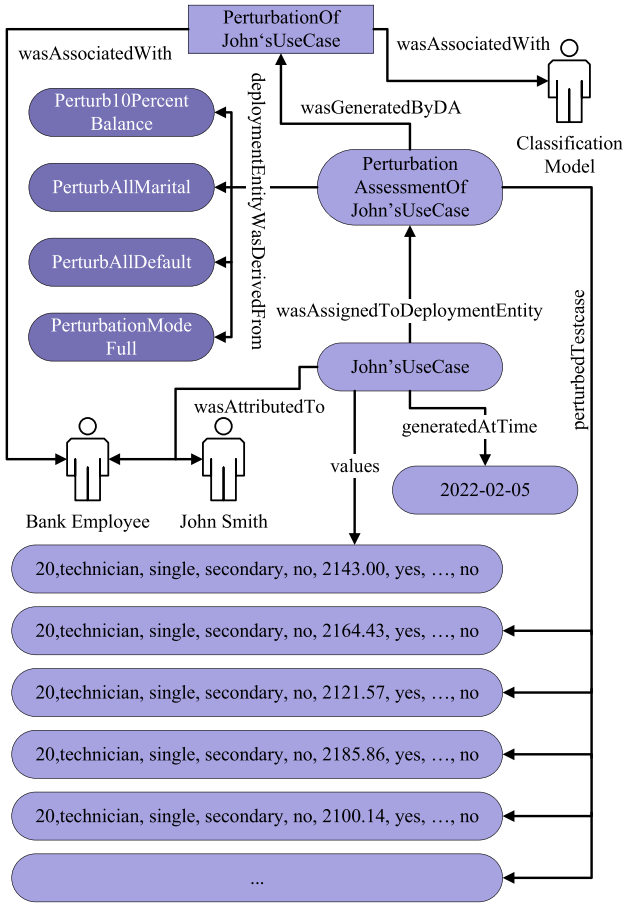

Fig. 17 Example extract of problem-specific level KG of deployment for telemarketing

assessment. The prediction returned by the indicated prediction model was assessed using three perturbation options. One for each of the three features balance, marital, and default. Further the employee can state that, for example, the balance was chosen for perturbation because it is a cardinal feature and has a medium volatility. Consulting a domain expert, the possible change for the balance was defined with ± 10%. It can also be stated, that the decision to use the *perturbation approach* to assess the reliability within the telemarketing case was taken on 24 January 2022 by an assigned manager.

Conclusions

This paper presented a generic reference process that facilitates the assessment of reliability of predictive analytics results; the reference process was demonstrated for classification problems. The CRISP-DM is the basis for the reference process, which enhances the CRISP-DM with reliability-related activities and entities in each stage.

Depending on the employed training data, different aspects of the data may serve to assess the reliability of an individual prediction made for a new input case. This paper specifically proposes two different approaches within the reference process, namely the use of local quality measures and the perturbation of input cases, which can be applied to assess the reliability. These approaches need to be defined and configured once for a data mining problem and can then

be used for each new prediction case to receive an assessment of the reliability for its prediction.

The proposed reference process is structured in three levels. The first level describes generic activities and collected metadata, represented by entities, and how they are connected together. Method-specific subclasses of the generic activities and entities are defined related to the used prediction method, illustrated in this paper using the example of classification. Then, the problem-specific level refers to an individual problem of the related prediction method.

A method-specific process can be instantiated for a specific problem, for example, using the method-specific process for classification to predict if one will make a long-term deposit or not. In order to properly configure and employ the reference process and all its related information we present the use of a knowledge graph based on PROV-O. The knowledge graph ensures the development of auditable reliability assessment processes and serves as storage for knowledge about any kind of information related to the assessment of reliability, for example, different method-specific processes or knowledge gathered through previous reliability assessment tasks. Obtaining additional knowledge about the reliability of predictions is not without effort but increased reliability of predictions supports decision-makers in critical business decisions.

Future work will focus on the application of the reference process for other prediction methods, e.g., regression or clustering, and different sorts of input data, e.g., natural language or images. Future work will also extend the list of assessment approaches to cover more aspects that can detect reliability issues for given predictions.

The proposed reference process can be used within various prediction problems, for example, we are currently working to assess the reliability of flight delay risk prediction for air traffic scenarios as presented in [25]. The classification model showed a global accuracy of 82.5% for the highest and, therefore, most important risk class, which should be predicted for the costliest air traffic scenarios. Assessing the reliability of individual predictions can be fruitful to identify false delay risk classifications and in this sense help to reduce costs that may be caused by these delayed flights. In order to be able to predict the flight delay risk, the model uses a huge amount of training data including historical flights, weather data, aircraft data, and notices to airman (NOTAMs). During business understanding, data understanding, and data preprocessing information about these data is gathered and can be used to assess the reliability of individual cases, for example, weather data is collected using a sensor with a defined precision, like the humidity at Atlanta airport. The *perturbation approach* can be used to assess the reliability of individual air traffic scenarios by perturbing humidity information within the sensor precision. The precision of the humidity sensor is documented during the data understanding stage of the data mining project. During the modeling stage, the information of the precision of the sensor can be used to create a perturbation option that perturbs a new humidity value within the precision range of the sensor. Does a prediction change while perturbing the humidity within the precision of the sensor, the reliability of the retrieved prediction should be further investigated by an analyst.

Funding Open access funding provided by Johannes Kepler University Linz.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no Conflict of interest.

References

1. Wirth R, Hipp J. CRISP-DM: towards a Standard process model for data mining. 2000; 11
2. Caetano N, Cortez P, Laureano RMS. Using data mining for prediction of hospital length of stay: an application of the CRISP-DM methodology. In: Cordeiro J, Hammoudi S, Maciaszek LA, Camp O, Filipe J editors. Enterprise information systems—16th international conference, ICEIS 2014, Lisbon, Portugal, April 27–30, 2014, Revised Selected Papers. Lecture Notes in Business Information Processing, vol. 227. Heidelberg: Springer; 2014. p. 149–166. https://doi.org/10.1007/978-3-319-22348-3_9
3. Moro S, Laureano R, Cortez P. Using data mining for bank direct marketing: an application of the CRISP-DM methodology 2011. EUROSIS-ETI
4. da Rocha BC, de Sousa Junior RT. Identifying bank frauds using CRISP-DM and decision trees. Int J Comput Sci Inf Technol. 2010;2(5):162–9.
5. Moro S, Cortez P, Rita P. A data-driven approach to predict the success of bank telemarketing. Decis Support Syst. 2014;62:22–31. https://doi.org/10.1016/j.dss.2014.03.001.
6. Cheney EW, Kincaid DR. Numerical mathematics and computing. Boston: Cengage Learning; 2012.
7. Moroney N. Local color correction using non-linear masking. In: The Eighth color imaging conference: color science and engineering systems, technologies, applications, CIC 2000, Scottsdale, Arizona, USA, November 7–10. 2000. p. 108–111. IS &T-The Society for Imaging Science and Technology, Springfield. 2000. http://www.ingentaconnect.com/content/ist/cic/2000/00002000/00000001/art00021

8. Capra A, Castorina A, Corchs S, Gasparini F, Schettini R. Dynamic range optimization by local contrast correction and histogram image analysis. In: 2006 digest of technical papers international conference on consumer electronics. Las Vegas: IEEE; 2006. p. 309–310. https://doi.org/10.1109/ICCE.2006.1598434
9. Reimer U, Tödtli B, Maier E. How to induce trust in medical AI systems. In: Grossmann G, Ram S editors. Advances in conceptual modeling–ER 2020 Workshops CMAI, CMLS, CMOMM4FAIR, CoMoNoS, EmpER, Vienna, Austria, November 3–6, 2020, Proceedings. Lecture notes in computer science, vol. 12584. Heidelberg: Springer; 2020. p. 5–14. https://doi.org/10.1007/978-3-030-65847-2_1
10. Vriesmann Jr LM, Oliveira LS, Koerich AL, Sabourin R. Combining overall and local class accuracies in an oracle-based method for dynamic ensemble selection. In: 2015 International joint conference on neural networks, IJCNN 2015, Killarney, Ireland, July 12–17, 2015. New York: IEEE; 2015. p. 1–7. https://doi.org/10.1109/IJCNN.2015.7280340
11. Sahoo S, Lebo T, McGuinness D. PROV-o: The PROV ontology. W3C recommendation, W3C. 2013. https://www.w3.org/TR/2013/REC-prov-o-20130430/. Accessed Oct 2021.
12. Huynh TD, Tsakalakis N, Helal A, Stalla-Bourdillon S, Moreau L. Addressing regulatory requirements on explanations for automated decisions with provenance—a case study. Digit Gov Res Pract. 2021;2(2):16–11614. https://doi.org/10.1145/3436897.
13. Staudinger S, Schuetz C, Schrefl M. A reference process for judging reliability of classification results in predictive analytics. In: Proceedings of the 10th international conference on data science, technology and applications—DATA. Setúbal: SciTePress; 2021. p. 124–134, https://doi.org/10.5220/0010620501240134
14. Winter PM, Eder S, Weissenböck J, Schwald C, Doms T, Vogt T, Hochreiter S, Nessler B. Trusted artificial intelligence: towards certification of machine learning applications. CoRR. 2021. arXiv:abs/2103.16910.
15. Wing JM. Trustworthy AI. Commun ACM. 2021;64(10):64–71. https://doi.org/10.1145/3448248.
16. Barr ET, Harman M, McMinn P, Shahbaz M, Yoo S. The oracle problem in software testing: a survey. IEEE Trans Softw Eng. 2015;41(5):507–25. https://doi.org/10.1109/TSE.2014.2372785.
17. Chen TY, Kuo F-C, Liu H, Poon P-L, Towey D, Tse TH, Zhou ZQ. Metamorphic testing: a review of challenges and opportunities. ACM Comput Surv. 2018;51(1):4–1427. https://doi.org/10.1145/3143561.
18. Segura S, Fraser G, Sánchez AB, Cortés AR. A survey on metamorphic testing. IEEE Trans Softw Eng. 2016;42(9):805–24. https://doi.org/10.1109/TSE.2016.2532875.
19. Xie X, Ho JWK, Murphy C, Kaiser GE, Xu B, Chen TY. Testing and validating machine learning classifiers by metamorphic testing. J Syst Softw. 2011;84(4):544–58. https://doi.org/10.1016/j.jss.2010.11.920.
20. Saha P, Kanewala U. Fault detection effectiveness of metamorphic relations developed for testing supervised classifiers. In: IEEE international conference on artificial intelligence testing, AITest 2019, Newark, CA, USA, April 4–9, 2019. New York: IEEE; 2019. p. 157–164. https://doi.org/10.1109/AITest.2019.00019.
21. Moreira D, Furtado AP, Nogueira SC. Testing acoustic scene classifiers using metamorphic relations. In: IEEE international conference on artificial intelligence testing, AITest 2020, Oxford, August 3-6, 2020. New York: IEEE; 2020. p. 47–54. https://doi.org/10.1109/AITEST49225.2020.00014.
22. Cruz RMO, Sabourin R, Cavalcanti GDC. Dynamic classifier selection: recent advances and perspectives. Inf Fusion. 2018;41:195–216. https://doi.org/10.1016/j.inffus.2017.09.010.
23. Didaci L, Giacinto G, Roli F, Marcialis GL. A study on the performances of dynamic classifier selection based on local accuracy estimation. Pattern Recognit. 2005;38(11):2188–91. https://doi.org/10.1016/j.patcog.2005.02.010.
24. Guo H, Shi W, Deng Y. Evaluating sensor reliability in classification problems based on evidence theory. IEEE Trans Syst Man Cybern Part B. 2006;36(5):970–81. https://doi.org/10.1109/TSMCB.2006.872269.
25. Bardach M, Gringinger E, Schrefl M, Schuetz CG. Predicting flight delay risk using a random forest classifier based on air traffic scenarios and environmental conditions. In: 2020 AIAA/IEEE 39th digital avionics systems conference (DASC). 2020. p. 1–8. https://doi.org/10.1109/DASC50938.2020.9256474.
26. Woods KS, Kegelmeyer WP, Bowyer KW. Combination of multiple classifiers using local accuracy estimates. IEEE Trans Pattern Anal Mach Intell. 1997;19(4):405–10. https://doi.org/10.1109/34.588027.