Model-Driven Business Process Analytics

Falko Koetter, Andreas Wohlfrom, Désirée Graf
Fraunhofer Institute for Industrial Engineering IAO
and University of Stuttgart IAT
Stuttgart, Germany
Email: firstname.lastname@iao.fraunhofer.de

***Abstract*—Business process analytics helps companies to optimize their processes by identifying shortcomings and improvement potentials. However, existing approaches require either a homogenous process execution environment or expert knowledge, both of which many companies lack. In previous work, we introduced aPro, a model-driven architecture enabling process monitoring even in heterogenous environments. Utilizing the model-driven approach, this work presents a new out-of-the-box approach for business process analytics, allowing the business user to analyze the process within the familiar context of the model without needing to know details of the implementation or data mining techniques. We evaluate this approach using both a simulated and a real-life process.**

Keywords–business process management; model-driven architecture; business intelligence; data mining

I. INTRODUCTION

Continuously optimizing business processes is a necessity in todays' fast-changing market [1]. But to identify relevant changes deficits in business process performance need to be known. For this, business intelligence or business process analytics is used, which transforms data obtained during process execution into metrics, Key Performance Indicators (KPI) and finally insights [2].

In previous work, we developed aPro, a model-driven methodology for creating a service-based monitoring infrastructure for business processes [3]. In aPro, a process model is augmented by a so-called goal model containing data to monitor as well as KPI and process goals to be calculated. From this model all components of the infrastructure are automatically created, including monitoring data collection, processing [4], visualization [5] and storage [6]. Using automatically created stubs, these components can be integrated in any business process execution environment, even allowing monitoring of processes run in heterogenous environments, e.g., legacy systems. aPro uses complex event processing (CEP) to process monitoring data, calculate KPIs and detect goal violations in real-time. However, this monitoring only provides data, but does not give any insights in relationships and causes, e.g., for goal violations. A way to analyze process data and detect optimization potential is needed.

In this work, we present an approach for model-driven business process analytics. Based on the aPro architecture, business process analysis has to occur on a level familiar to the business user. As aPro is a model-based approach, the technical details of the implementation are hidden from the business user, who only works with the conceptual goal model. Thus, the analysis needs to present results on this level as well. to achieve this, we investigate exisiting approaches for data mining in regards to their suitability for model-driven business process analysis. In particular, no detailed configuration or expert knowledge of underlying techniques is to be required from the business user. We select two suitable techniques and implement them within the apro prototype. The approach is evaluated using a synthetic and real life use case.

The remainder of this work is structured as follows. In Section II, we give an overview of related work. In Section III, we detail the concept for model-driven business process analytics, including the architecture, extensions and requirements. Section IV contains the selection of data mining techniques sutiable for the requirements. We shortly describe the implementation in Section V. We evaluate the work using both a synthetic and real-life process in Section VI. In Section VII, we give a conclusion and outlook of future work.

II. RELATED WORK

In this paper, we describe a technique for model-driven business process analytics. This encompasses the following topics of related work: Model-driven business process

One of the design criterions is integration within the aPro methodology for model-driven process monitoring. In aPro, the process goal modeling (ProGoalML) language is used for modeling the monitoring of a business process, including measurements, KPIs and goals [3]. Measurements are taken at run-time and correlated by process instance using CEP. From the correlated measurements, KPIs and goals are calculated, either for a single process instance or for an aggregation of process (e.g., all instances of the last hour). While this monitoring allows for basic analysis by visualizing data in charts and giving alerts in case of goal violations, it is not sufficient for long-term analysis. As a preliminary work we extended aPro with a data warehouse, which is automatically configured from a ProGoalML model [6]. It provides data for business process analytics.

As model-driven business process analytics aims to be an out-of-the-box technique, related work exists in the areas of business process intelligence and business process analytics.

Castellanos et al. [7] describe one of the first tools for out-of-the-box business process intelligence. Business processes are modeled in a specific notation and executed in a process engine. Process metrics, similar to KPIs and goals in aPro, are modeled in forms and calculated from process audit logs during extraction to a data warehouse. Data mining techniques like decision trees and classification algorithms are used to generate insights, which are visualized in a cockpit, from which process execution can be controled as well. In comparison to this work, metrics are defined separately from the process model, process execution is limited to a process engine and result visualization is separated from the process model as well. In comparison, aPro aims at a high degree of integration between process modeling, goal modeling and analysis, as well as supporting a wide range of execution environments.

 ISBN: 978-1-61208-412-1

The *deep Business Optimization Platform* (dBPO) is a platform for business process optimization before, during and after execution [8]. Before execution the process is optimized investigating the process model and finding structural optimization potentials, e.g., by cross-referencing patterns of best practices [9]. During execution optimization may be changing parameters, e.g., switching web-services. After execution, processes may be adapted using insights gained from execution data [8]. To gather this data, dBPO uses a semiautomated mapping connecting execution data from a process engine and operational data from heterogenous sources. Using this data, parts of the process with high potential for optimization are identified. Using customized data mining techniques, the applicability of patterns is tested, e.g., testing if an ability can be split into distinct variants. These patterns can be applied semi-automatically to the process model depending on the goals set by the analyst. The approach has been implemented using the Business Process Execution Language (BPEL). While dBPO offers sophisticated optimization techniques, it requires a BPEL compliant execution environment as well as operational data from an independent source. In comparison, aPro provides its own method for capturing operational data (i.e., monitoring stubs), which is integrated ex-ante instead of matching data ex-post. Data is then captured as part of process execution and stored in an automatically configured data warehouse, making independent data storage unnecessary.

Similarly to aPro, Wetzstein et al. [10] present a business process intelligence system for monitoring process goals, which are derived from process metrics. Processes are executed in BPEL, monitoring data is gathered by monitoring the services orchestrated within the process. Metrics and goals are stored in a metrics database, which provides data for an analysis step. During the analysis, a decision tree is generated for each process goal depicting the influential factors leading to goal fulfillment or violation. In comparison to aPro, the approach is limited to BPEL processes and the metrics, which can be derived from service calls.

Overall, the related work shows a high degree of maturity in data mining techniques as well as highly integrated approaches covering the complete business process lifecycle. However, in all cases expert knowledge in some domains of the lifecycle is needed. Therefore, we decided to use existing data mining techniques, but provide a new approach for analytics on the modeling level, hiding the implementations of other parts from the business user.

III. CONCEPT

In this section, we summarize the existing aPro architecture, define the requirements for analytics within this context and describe the concept for model-driven business process analytics.

A. aPro overview

Model-driven process analytics build on the established aPro architecture, which is shown in Figure 3. Conceptually, the system is divided. The *model layer* represents a business view of the process and its monitoring. The underlying implementation of the process is hidden. The *aPro layer* contains automatically configured components of the aPro architecture for monitoring. The *execution layer* contains the concrete implementation of the process, which consists of one or more *executing systems*.

On the model layer a *process model* and a *goal model* containing metrics, KPIs and goals are created in a graphical editor, as shown in Figure 1. The process model depicts a simplified real-life claim management proess, as is performed by an insurance service provider many times a day. When a damage claim (e.g., for a damaged vehicle) is received from an insurance company, it is checked using rules to independently calculate the claimed amount. It then is decided, if the claim is valid and what amount is paid out. In the final step, the results are sent back to the insurance company.

Below the process model a goal model is shown in the lower half of Figure 1. The most important elements of the ProGoalML notation are shown in Figure 2. Attached to each process activity is a *measuring point*, indicating a measurement is taken during execution of this activity. It contains the *parameters* to be measured, which are named values of a specific data type. For example, the measuring point *receive_mp* is attached to the activity *receive_claim* and contains three parameters: a *case_id* of type *ID* identifying the current case, a *timestamp (TS)* indicating the time the claim is received and a *claimed_amount* of type *Double (D)* in Euro. Other parameter types are *String (S)*, *enumeration (E)* and *Boolean (B)*.

KPIs are connected by arrows to the parameters (or other KPIs) they are calculated from. For example, the KPI *savings_percentage* is calculated from the parameters *claimed_amount* and *calculated_amount*. Similar to parameters, KPIs have a specific data type. Process goals may be imposed on KPIs or parameters and determine the success of process executions. A goal is a Boolean condition on a KPI or parameter. If it is not fulfilled, compensating actions may be triggered [11]. For example, the goal *five_percent_saved* is imposed on the KPI *savings_percentage*. A special type of goal is a *timing_goal*, which is fulfilled if the time between two measurements is below a specified value. In the example, the timing goal *time_sla* is fulfilled, if a claim is processed within 15 seconds.

These models are stored in a ProGoalML file, which is used as basis for the model transformation. In this step, the components of the aPro architecture are created and/or

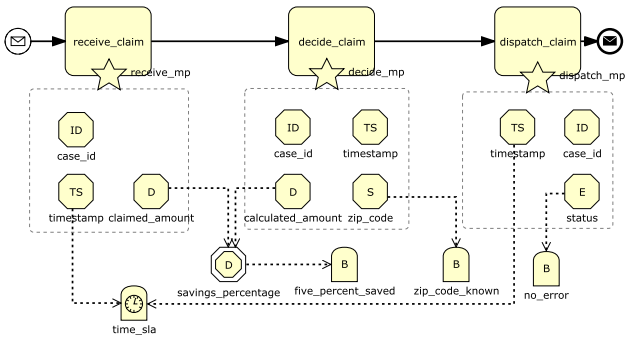

Figure 1. Example process and goal model: claim management process

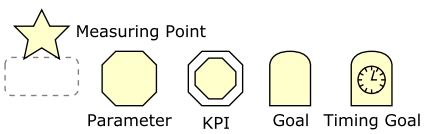

Figure 2. Overview of ProGoalML elements [3]

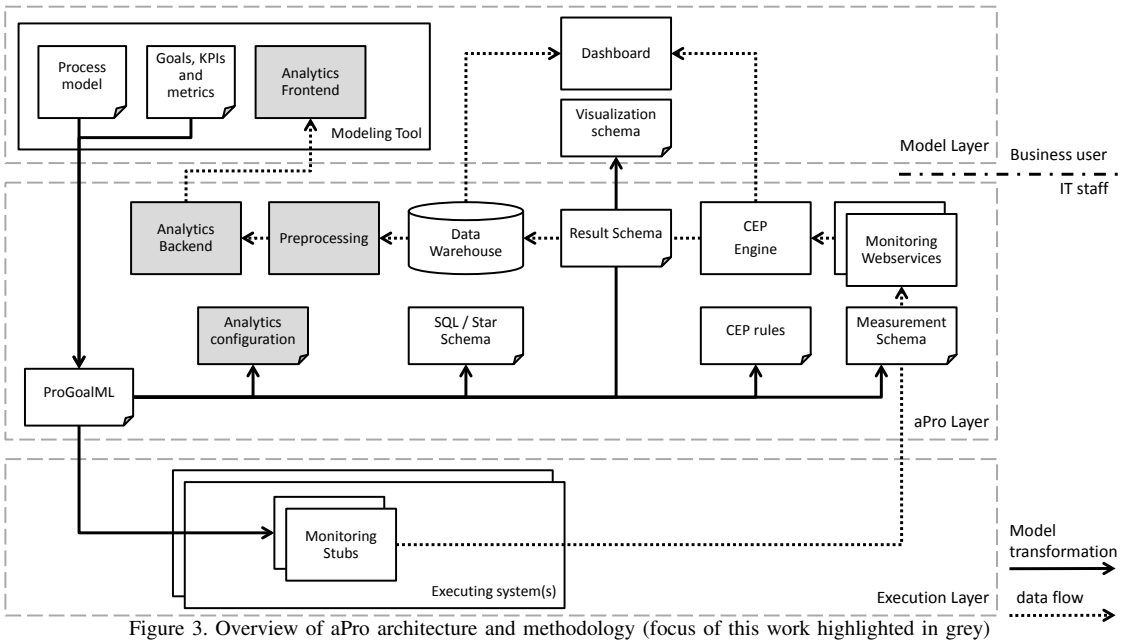

Figure 3. Overview of aPro architecture and methodology (focus of this work highlighted in grey)

configured. Monitoring stubs are created, which are integrated automatically (e.g., in a process engine) or manually (e.g., in a java webservice) in the executing systems to gather measurements. These measurement are sent to *monitoring webservices*, which in turn deliver them to a *CEP engine* for processing. *CEP rules* to correlate measurements of process instances as well as calculate KPIs and goals are generated automatically. One of the results of CEP is an XML file (*result schema*) containing all parameters, KPIs and goals of a process instance. These files are imported into a *data warehouse* for long-term storage. Short-term data from the CEP engine and long-term data from the data warehouse are visualized in a *dashboard*, which is configured using an automatically generated *visualization schema*.

While data visualization is sufficient to monitor system operations, to optimize the process a deeper survey of the data is needed. In the course of this section, we will formulate the requirements as well as the concept for model-driven process analytics.

B. Requirements

The main goal of business process optimization with aPro is the fulfillment of process goals. Thus, the goal of business process analytics is to determine which circumstances lead to the violation of a goal. These circumstances may not readily apparent and cannot be derived in an analytical way from the process model. This suggests the use of knowledge discovery [12] to derive these hidden dependencies from execution data. However, different data mining techniques used in knowledge discovery have several advantages and drawbacks and have to be selected for each problem individually. In this section, we detail the requirements for business process analytics in the context of aPro. These requirements will be used to (a) create a concept and (b) select suitable data mining techniques.

1) **Influential factors to goals** To get a deep insight into the process the influential factors of a goal need to be identified. Which factors aid or hinder goal fulfillment to which extent?
2) **Types of data dimensions:** The data dimensions of measurements and KPIs are not uniformly scaled. The more scales (nominal, ordinal, interval, ratio or absolute) the better the concept is suited.
3) **Model layer presentation:** Due to the model-driven approach, the execution and aPro layers are invisible to the business user. Thus, the analysis results must be contextualized within the process and goal model.
4) **Intuitive comprehensibility:** As a business user is not a data mining expert, the used techniques have to provide results which are comprehensible to him or her.
5) **Visualization of results:** Results need to be visualized in dashboards, diagrams, reports, etc. in a user frontend.
6) **Automatisation:** As the business user is not qualified to adjust parameters, the chosen techniques have to provide acceptable results with default or automatically derived parameters.
7) **Accuracy:** As the target attribute is Boolean the simple rate of correct classified samples is used as a measure for accuracy.
8) **Data warehouse compatibility:** The input data is provided by the data warehouse. The solution has to be compatible with this data structure with only automatic conversions.

C. aPro extension

According to [13], knowledge discovery consists of four steps. During *cleaning and integration (1)*, data is gathered from different sources, correlated and data sets with errors or gaps are corected or excised. During *selection and transformation (2)*, redundant or irrelevant attributes are removed and data is converted in a format fit for *data mining*. Especially, numerical values may be transformed to nominal values, for example by summarizing them to intervals. This process is called discretization and is used because some data mining techniques only work on nominal values [14]. During *data mining (3)*, patterns and insights within the data are discovered.

 ISBN: 978-1-61208-412-1

These are presented to the user during *knowledge presentation (4)*.

Adapting these steps for model driven process analytics within aPro shows that *cleaning and integration* is already performed by existing components. *Monitoring webservices* accept measurement data only in a defined schema sent by *monitoring stubs*. The *CEP* correlates measurements to results spanning process instances, generating complete and unified data sets.

To perform the other steps, additional components are added to the aPro architecture (Figure 3, highlighted in grey).

A *Preprocessing* component reads data from the data warehouse and performs the *selection and transformation* step. Ordinarily, during data selection incomplete datasets are excised. However, in aPro, missing data is not caused by a lack of record-keeping, but rather by a lack of measuring, e.g., if a measuring point is attached to an optional activity. In this case, a lack of measurement indicates the optional activity has not been performed. Thus, missing values are explicitly set to *no_value*. During implementation, we discovered another property of monitoring data in aPro. As goals are always derived from a single attribute (either a parameter or KPI), that attribute is a sufficient influential factor. It alone can be used to determine goal fulfillment. To avoid this problem, during selection all attributes the target attribute is directly derived from are excised. The attributes as well as their data types and dependencies are known from an *analytics configuration*, which is automatically generated from ProGoalML.

The preprocessed data is provided to the *Analytics Backend*. Here, the *data mining* step takes place. Taking into account the requirements defined above, suitable data mining techniques are selected in the following section.

The Analytics Backend offers the implemented data mining techniques as a webservice. That webservice is called by the *Analytics Frontend*, which is integrated in the modeling tool. This allows for a model layer presentation of results, for which the frontend offers a range of visualizations.These visualizations can visualize results from any technique, as long as they are in the correct data format, thus enabling extensibility.

1) **Colorization:** Colors elements of the process and goal model along a gradient. A red/green gradient ist used to visualize relative information gain of metrics, KPIs and goals in relation to a selected goal.
2) **Picture:** Displays a picture rendered by the backend, for example generated decision trees. A picture is linked to a model element. It will be shown when this model element is selected.
3) **InfluenceChart:** A bar chart displaying the top influence factors of a selected goal. In case of the naive Bayes classifier, influence factors are ranked by relative information gain.
4) **DistributionChart:** A bar chart showing the distribution of process instances among values of a selected parameter or KPI (i.e., how many instances have value x) or the distribution of a single value in regards to a selected goal (i.e., how many instances with value x fulfill a goal or not).

The *Analytics Frontend* offers an analysis mode, which locks the process model for edits. Now, one or more techniques for analysis can be chosen. For these techniques, calls to the Analytics Backend are created from the data. As soon as result data is provided, modeling elements can be selected to show results. For example, the selection of a goal can show a picture of a decision tree or color the goal model according to relative information gain from naive Bayes classifier. Visualizations are organized in a dashboard, which allows displaying multiple visualizations side by side and moving them around the model.

After analysis is finished, analytics mode can be deactivated and the model can immediately be adapted.

IV. Selection of data mining techniques

Modern computer science offers a wide variety of data mining concepts. As the target variable is a goal, it is known for all instances. Therefore, process analytics in aPro is a supervised learning problem. For the selection, we chose six data mining techniques to evaluate according to the requirements introduced in section III-B except model layer presentation. These techniques are widely used and described in several works [14][15].

A. Decision trees

Decision trees use the entropy of the different influential factors to build up a sequential classification. A drawback of decision tree is the intransparency of the dependencies between the attributes [16]. The most important advantage of decision trees is the intuitive understanding of decision trees and there high grade of accuracy and automatisation [17].

B. Support vector machines

Support vector machines seperate the data set in two classes by maximizing the distance of the support vectors. Support vector machines are easy to understand if the kernel function is linear. If a non-linear kernel function is used, understanding of results and dependencies is hard. However support vector machines provide high accuracy and can handle with all scales [18].

C. Bayesian networks

Bayesian networks are directed acylic graphs and for each vertice is a conditionally random distribution calculated. Bayesian networks provide a high transparency of dependencies and an intuitive visualization. However, the requirement for automatisation of the analysis is not fulfilled and thus not suitable for business users [18].

D. Naive Bayes classifiers

Naive Bayes classifiers classify the data due to their marginal distributions with the condition that the influential factors are stochastically independent. The assumption that the attributes are stochastically independent is leading to a classifier that fullfills all requirements. The misclassification increases with that assumption but we found the correct classification rate to be still sufficient. Another advantage of the naive Bayes classifier is the intuitive understanding [18].

E. Neural networks

Neural networks have few layers with perceptrons in each layer and the perceptrons transmitting signals through the neural network to activate other perceptrons. Neural networks with their black-box-system make it impossible to identify the dependencies between the target variable and the influential factors. On the other hand, neural networks have a high accuracy [19].

 ISBN: 978-1-61208-412-1

TABLE I. REQUIREMENT FULLFILLMENT OF EVALUATED MACHINE LEARNING CONCEPTS.

Requirements \ Concepts	Decision tree	Support vector machines	Baysian networks	Neural networks	Naive Bayes classifier	Lazy learner
Influence factor dependency	+	o	+	-	+	-
Types of data dimension	o	+	+	+	+	+
Intuitive understanding	+	-	o	-	+	-
Visualization	+	o	o	o	+	o
Automatisation	+	-	o	o	+	o
Accuracy	o	+	+	+	o	o
Data warehouse compability	+	+	+	+	+	+

F. Lazy learners

Lazy learner save the whole instances and compare every new instance with the saved instances. Since lazy learners classify instance-based the technique is comprehensible and all scales can be used. However, to identify dependencies between the attributes and the target variable lazy learning is unsuitable.

For each requirement and each classifier we evaluated if the classifier fulfills the requirements fully (+), partially (o) or not at all (-). The table I shows the results of this evaluation. Taking into account the results we chose decision trees and naive Bayes classifiers for the implementation. There are several algorithms that generate decision trees. We chose C4.5 [20] as an algorithm which creates small decision trees to aid intuitive understanding. However, we kept the approach extensible in case other techniques are needed.

V. IMPLEMENTATION

We extended the existing aPro prototype [4] to encompass model driven business process analytics as shown in Figure 3.

In the analytics backend, we utilize WEKA [21], which contains reference implementations of data mining techniques. The preprocessing component reads data from the data warehouse according to the structure implied by the ProGoalML file and converts it to WEKA-compliant input formats.

The frontend is based on the Oryx Editor [22] and existing dashboard components for visualization. Figure 4 and Figure 5 shows the analytics frontend with results from simplified real-life data.

VI. EVALUATION

We will evaluate the approach using two business processes, evaluating (a) if the chosen algorithms provide suitable results and (b) if the visualizations can communicate these results. The first process is a blood donation process [23] and the second is a real-world claim management process. The first process was the training process evaluated using synthetically generated data. The data set of the blood donation process has seven dimensions and one target variable. Data was generated by a test data generator. For the second process we had 48 dimensions in the data set, one target variable and 5718 cases. We used analytics to find the dependencies hidden in the generated data sets. The second process we evaluated with anonymized real-life data as the test process. As a metric we used the plain accuracy and made a 10 cross-validation. The different techniques are validated with the differently prepared

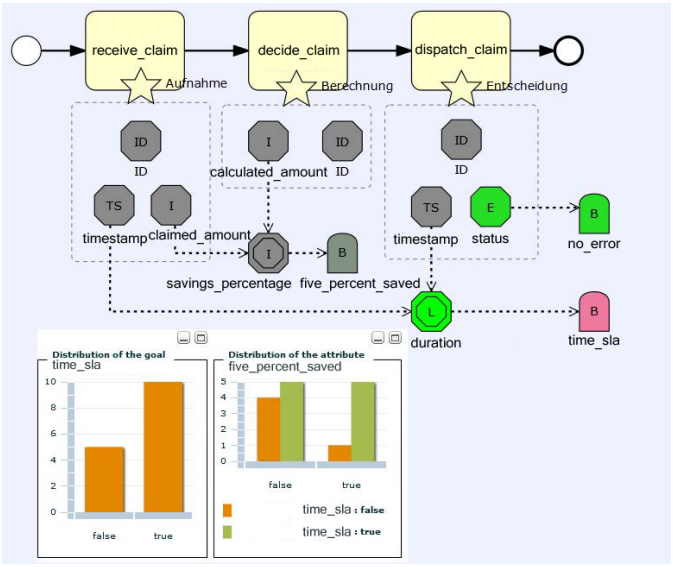

Figure 4. Analysis of example process with simplified real-life data (translated from German)

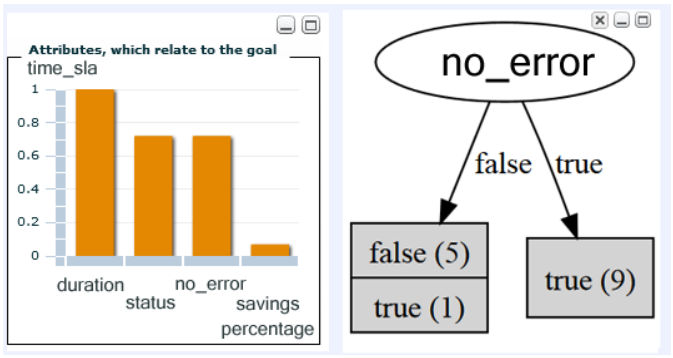

Figure 5. Influence factors and decision tree for the goal time_sla (translated from German)

data sets III-C. Learning accuracy was evaluated by 10-fold cross validation.

A. Blood donation process

Most of the generated dependencies were found by the naive Bayes classifier. Other found dependencies were not explicitly generated but found to be emergent from the process simulation. The extreme cases (direct dependency or random numbers) were correctly classified. The accuracy of the classifier was high (87%). We modified the generation of the data set and the information gain reflected all modifications. It was possible to identify the strong dependencies with a high information gain. The increase or decrease of the information gain is a stable and robust measure. The decision tree found the direct KPIs or parameters for a goal. Therefore it was necessary to exclude these KPIs and parameters as described in Section III-C. After the exclusion of these KPIs and parameters the accuracy increased. Using discretization the tree size shrank without a significant decrease in the accuracy. The accuracy of the decision trees was less high (61%). Reasons for this low classification rate could be the low base line or the absence of high dimensions in the data set.

B. Claim management process

The discretization of the data set increased the accuracy of the naive Bayes classifier. If the data set was prepared

the classification rate was slightly better. Overall the highest accuracy of 96,1% of the naive Bayes classifier occurs with a discretization without preselection. The best results with decision trees were with a discretization together with a information gain ratio data selection and a discretization without a selection. We use this configuration as a default in the prototype. The worse performance of the decision trees arised from the low number os attributes in the blood donation process data. The real world claim management process has hundreds of parameters, KPIs and goals. We evaluated rules for fraud and irregularity detection and retired over half of the rules with low information gain as well as found several new rules from decision trees, evaluating an anonymized dataset of 98167 cases.

The naive Bayes classifier fulfills all requirements and has an accuracy of 87% in the synthetically generated data set. The accuracy in the real-world data set even higher (96,1%). Furthermore the information gain is a robust and stable measure for dependencies in aPro. Decision trees as an analytics concept are only partially suitable for aPro. Decision trees represent the most strongest dependencies and neglect other influential factors because it is sufficient to classify the instances with these stronger dependencies. The accuracy of decision trees in aPro was 61% in the synthetical data set and 90,7% in the real-world data set.

Overall the combination of decision trees and naive Bayes classifier are suitable for model-driven process analytics because decision trees allow a quick overlook over the most strongest dependencies in the data sets and the naive Bayes classifier complements this overview with high accuracy and deeper insights into the dataset.

VII. Conclusion and outlook

In this work, we presented an approach for model-driven business process analytics using the aPro architecture. The main contribution is the support of diverse process execution environments, while still providing out-of-the-box analytics for business users, hiding both monitoring and implementation layers. We defined criteria for data mining techniques and selected two exemplary techniques. The concept proved sufficient during evaluation and real-world use. However, support for more data mining techniques and different configurations may provide better insights, while still preserving ease-of-use for business users.

In future work, we would like to extend the approach to process adaptation as well as compliance scenarios. As process goals can be used to monitor compliance requirements, reporting and root cause detection for compliance violations can be provided [24].

Acknowledgment

The work published in this article was partially funded by the Co.M.B. project of the Deutsche Forschungsgemeinschaft (DFG) under the promotional reference SP 448/27-1.

References

[1] N. Slack, S. Chambers, and R. Johnston, Operations management. Pearson Education, 2010.

[2] M. zur Mühlen and R. Shapiro, "Business process analytics," in Handbook on Business Process Management 2. Springer, 2010, pp. 137–157.

[3] F. Koetter and M. Kochanowski, "Goal-oriented model-driven business process monitoring using progoalml," in 15th BIS. Vilnius: Springer, 2012, pp. 72–83.

[4] ——, "A model-driven approach for event-based business process monitoring," Information Systems and e-Business Management, 2014, pp. 1–32.

[5] M. Kintz, "A Semantic Dashboard Description Language for a Process-oriented Dashboard Design Methodology," in Proceedings of 2nd International Workshop on Model-based Interactive Ubiquitous Systems (MODIQUITOUS 2012), Copenhagen, Denmark, 2012.

[6] F. Koetter, M. Kochanowski, M. Kintz, T. Renner, and P. Sigloch, "Model-driven data warehousing for service-based business process monitoring," in Global Conference (SRII), 2014 Annual SRII. IEEE, 2014, pp. 35–38.

[7] M. Castellanos, F. Casati, U. Dayal, and M.-C. Shan, "A comprehensive and automated approach to intelligent business processes execution analysis," Distributed and Parallel Databases, vol. 16, no. 3, 2004, pp. 239–273.

[8] F. Niedermann and H. Schwarz, "Deep business optimization: Making business process optimization theory work in practice," in Enterprise, Business-Process and Information Systems Modeling. Springer, 2011, pp. 88–102.

[9] F. Niedermann, S. Radeschütz, and B. Mitschang, "Business process optimization using formalized optimization patterns," in Business Information Systems. Springer, 2011, pp. 123–135.

[10] B. Wetzstein, P. Leitner, F. Rosenberg, I. Brandic, S. Dustdar, and F. Leymann, "Monitoring and analyzing influential factors of business process performance," in Enterprise Distributed Object Computing Conference, 2009. EDOC'09. IEEE International. IEEE, 2009, pp. 141–150.

[11] F. Koetter, M. Kochanowski, M. Kintz, and I. Fraunhofer, "Leveraging model-driven monitoring for event-driven business process control," in 1. Workshop zur Ereignismodellierung und verarbeitung im Geschaeftsprozessmanagement (EMOV), 2014, pp. 21–33.

[12] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth et al., "Knowledge discovery and data mining: Towards a unifying framework." in KDD, vol. 96, 1996, pp. 82–88.

[13] J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011.

[14] I. H. Witten and E. Frank, Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2005.

[15] S. Theodoridis and K. Koutroumbas, "Pattern recognition," IEEE Transactions on neural networks, vol. 19, no. 2, 2008, p. 376.

[16] S.-M. Lee and P. A. Abbott, "Bayesian networks for knowledge discovery in large datasets: basics for nurse researchers," Journal of Biomedical Informatics, vol. 36, no. 4, 2003, pp. 389–399.

[17] S. Tufféry, Data mining and statistics for decision making. John Wiley & Sons, 2011.

[18] C. M. Bishop et al., Pattern recognition and machine learning. springer New York, 2006, vol. 1.

[19] J. Schmidhuber, "Deep learning in neural networks: An overview," CoRR, vol. abs/1404.7828, 2014, last accessed 29.01.2015. [Online]. Available: http://arxiv.org/abs/1404.7828

[20] J. R. Quinlan, C4. 5: programs for machine learning. Morgan kaufmann, 1993, vol. 1.

[21] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," ACM SIGKDD explorations newsletter, vol. 11, no. 1, 2009, pp. 10–18.

[22] G. Decker, H. Overdick, and M. Weske, "Oryx — An Open Modeling Platform for the BPM Community," in Proceedings of the 6th International Conference on Business Process Management. Heidelberg: Springer, 2008, pp. 382–385.

[23] D. Schleicher, C. Fehling, S. Grohe, F. Leymann, A. Nowak, P. Schneider, and D. Schumm, "Compliance domains: A means to model data-restrictions in cloud environments," in Enterprise Distributed Object Computing Conference (EDOC), 2011 15th IEEE International. IEEE, 2011, pp. 257–266.

[24] F. Koetter, M. Kochanowski, A. Weisbecker, C. Fehling, and F. Leymann, "Integrating compliance requirements across business and it," in Enterprise Distributed Object Computing Conference (EDOC), 2014 IEEE 18th International. IEEE, 2014, pp. 218–225.

 ISBN: 978-1-61208-412-1