This work is contextualized in the Process Mining. Wich is a young discipline providing several techniques for Business Process understanding, where a process can model a real situation from world or can specifies for example a software procedure.

In this work the focus is put on Process Analysis that assumes the presence of a Process model and an event log in wich all the process executions are traced.

Let us start by presenting a running example of Business Process. This is a fragement of a sale procedure showing how the orders received are handled.
At the top of the figure there is the BPMN model of the process. After a first activity in which the order is simply received and registred, they follow some evaluation activities performed in parallel. In particular the financial checking takes into exam the financial situation of the customer to verify his reliability.
The warehouse activity instead checks if the are enough merchandise to serve client, and in case of a shortage an external warehousing activity is started. The final activity is for notifying the order result to the customer.
At bottom we find the petri net model which can be obtained from BPMN model using a transformation techniqe. We need also a Petri net model since there are a lot of existing analysis techniques that consider Petri net instread of BPMN.

Now given a model of a process representing the “ideal” process behavior, and given an event log in witch the real behavior is traced: Process Analysis takes into exam these two elements and analyse ander a conformance and performance points of view.

Conformance analysis checks if a trace or a process execution is compliant with the process model, while performance checking takes into exame the performance level of the process for example through the computation of some metrics related to the execution time synchronization time in case of parallel branches of the workflow and so on..

Both conformance and performance analysis are done thanks to an algorithm called log replay.

Log replay assumes that an event log is a finite sequence of traces. A trace is a finite sequence of events and it concerns a single process execution. While an event can be seen as a couple: action which tells what is the activity performed, and timestamp of the event. We consider also that an activity can have only two possible states: start and complete. In addition we assume that each event can be mapped into a transition of the Petri net model.

Log replay algorithm takes as input the Petri net model of the process and an event log. For each trace it starts by placing a token in the start place of the net, extracts the event in head of the trace and fires the corresponding transition in the net. If the transition to be fired is not enabled, log replay creates the missing tokens artificially so that can conlude the replay of the entire trace.

At the end, all the missing tokens created artificially can be analysed for the conformance checking.

In fact, if we consider the trace in the figure related to the sale business process and its Petri net model, the log replay execution is blocked in correspondence of the event Notify_start, this is because the transition t_10 is not enabled, In fact the marking reached by the net includes one token in P5 and one token in P10. In order to enable transition 10, a token should be present in the place P11. But to have this marking the transition financial_check_complete should be fired and this can be done only if the log replay extracts the event financial_checK_complete, wich is not present in our trace. So to complete the replay, the algorithm creates artificially a token in P_12 to enable directly the transition that must be fired, that is Notify_start.

The missing token created signals that something wrong is gone, in fact the trace is not conforming to the model since the communication to the customer had been performed before the completion of the financial evaluation activity.

Well, now we move to the real contribution of our work which is considering the possibility of exploiting all the data recorded in the event log which is not considered by the log replay technique. To do this, we consider machine learning techniques.

In general, exploiting ML discipline for the business process understanding is an approach already considered.. for example in extracting a process model based on event logs, or in discovering rules associated with decision points of the process..

Instead our idea is to explore how the process data may influence its behavior from a conformance point of view.

In particular, Conformance checking can be seen as a classification problem.
A process execution can belong to two classes: the class of conforming executions and the class of not conforming ones, and a conformance classifier is enable to assigns to each execution its class label witch is the conformance result.

Transforming the conformance checking in this way enable us to identify the possible causes of non-conformance. This is because the classifier model detects rules and patterns on data in correspondence of witch conformance errors occur.

From an other point of view, having a conformance classifier sometimes allows to predict the conformance result of a process execution even before the completion of the entire execution. For example, if all the attributes needed for the classification are known before the process termination. And this can be powerfull because in case of possible errors, process actors could take the particular instance carefully in order to avoid the errors.

All these purposes can be reached handling the conformance problem as a classical classification problem, so learning from experience, in our case that means learning from the previous process analysis. But in order to understand the conformance rules, we should employ an explicit classifier, in our case we consider the Decision Tree model.

In this contribution we propose a simple approach consisting of 4 steps for handling conformance analysis as a classification problem.

First we have to collect a data set for the learning. This is done based on an event logs and their conformance result computed with the log replay. So we have a data set in which each record is given by data recorded in corrispondence of a single execution of the process, and having as target attribute the conformance result of the trace.

Once a data set is formed they can follow some preprocessing activities, for example a feature selection. Now the step of learning can take place and a decision tree model can be learned through a machine learning algorithm.

The last step proposed consists in using the classifier constructed, we can use the decision tree in two ways: exploring the rules detected by the decision tree in order to understand conformance errors causes, and using the DT by classifying new process instances.

Now, let's consider the sale business process to illustrate this approach.

We take an event log L of the sale business process and we execute the log replay. The labeled Petri net sohws that 37 traces don't respect the process model, and this is because the financial evaluation activity wasn't performed for these traces.

Now the next step is to merge the conformance results with the process data extracted from the event log L.
The table on the figure shows the dataset constructed, in this case we focus on the process activity actors.. for example for each process instance we have an attribute about the costumer typology if he is a new client or a consolidated one, an attribute about the sales manager who handles the entire process, a financial officer who performes the evaluation activity and so on..

Learning based on this data set results in the decision tree shown on the figure.
The rule detected for non-conformance tells us that orders issued by consolidated clients of the organization and evaluated by the financila officer Mary may not respect the standard sale procedure.

So what can we do with this information? For example, if it turned out that is reasonable that for consolidated clients it is not necessary to perform financial evaluation, we can for example extend the model introducing a decision point after an order reception.. Or we can decide just to take some simple corrective measure.

In order to do some experiment of this approach, we realized an analysis framework started from ProM which is a process mining open source software, providing several tools for handling business processes and their analysis. It is written in Java and can be extended with other tools as plugins.

And this is exactly what we did integrating Weka framework for the ML part.

We realized plugins that work in a pipline. The figure shows dependecies between them. Their are mainly 4 important plugins corresponding to the 4 steps of the approach that I illustrated. Fo example the plugin Weka instances with conformance is for the data set creating, the generate weka classifier plugin is for the decision tree learning and so on..