

# Discrimination Aware Process Analysis Plugin Manual

Mahnaz Sadat Qafari

Chair of Process and Data Science, RWTH Aachen University, Aachen, Germany  
`m.s.qafari@pads.rwth-aachen.de`

**Abstract.** Process mining is a multi-purpose tool enabling organizations to monitor and improve their processes. One of the primary purposes of process mining is finding the root causes of performance or compliance problems in processes. However there are many factors influencing a process and therefor root cause analysis using approaches based on correlation and pattern detection are prone to making obvious or unfair diagnoses. Moreover, applying the results of these approaches may result in conclusions that are unsurprising or even discriminating. In this paper, we present a tool for solving this problem by creating a fair classifier for such situations. The undesired effects are removed at the expense of reduction on the accuracy of the resulting classifier.

## 1 Introduction

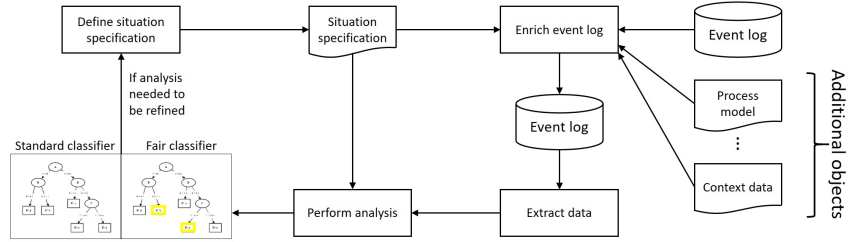
Academic and commercial process mining tools aim to find the root causes of performance or compliance problems in processes. Mainly, a classifier, say a decision tree, is created using the data gathered from the process and then the rule mining is done using that decision tree. However, this approach may lead to diagnoses that are not valuable. In some cases, the main cause of the problem is already known and essentially cannot be altered. Also, due to the strong correlation of the known main cause and the problem, it may become impossible to see the other minor but probably more practically valuable causes of the problem. Consider the following two scenarios: (i) there is a bottleneck in the process and it is caused by the busiest employee, or (ii) there are deviations caused by the most experienced resources taking the most difficult cases. In these scenarios, it is likely that the busiest employees or the most experienced resources are declared the main reasons for the bottleneck or deviations in the process. This is not just unfair but also does not provide novel insights (just stating the obvious). Even if we remove the attribute conveying the employee or the resource, still rules that proxy these attributes would be revealed. In these cases, it is essential to make inference about the less trivial root-causes of the problem in the process.

One way to solve such problems is removing the correlation between the feature that effect the problem but its effect is acceptable (sensitive feature) and the problem (class feature). We formulate the situation as a discriminatory

case where we aim to create a fair classifier where the correlation between the sensitive feature and the class feature is less than a specific amount  $\epsilon \in [0, 1]$ . The we use this classifier for the root cause analysis. To generate a fair classifier (here a decision tree), first we build a standard decision tree and measure the level of discrimination in it, if it is higher than  $\epsilon$ , the unacceptable discrimination (correlation) is removed via relabeling technique. Please refer to [1] for more information.

As another application, consider that for a given process we are interested in questions which are related to investigating the process while ignoring the effect of different values of a particular attribute. “Following the progress of career paths while eliminating gender differences” is one example of these sorts of situations where we need to remove the correlation between two attributes in the data.

For a given situation specification, we go through the following three steps: (1) Enriching the log, (2) Extracting the data, and (3) Learning fair classifier. The general approach of our method is depicted in Figure 3.



**Fig. 1.** The general framework proposed for fair root-cause analysis. First, according to the situation specification the event log is enriched by preprocessing the log and other sources of information. Then, the data is extracted from the enriched event log. Finally, two standard and fair classifier are created. Based on the analysis result, it is possible to adapt the situation specification to gather additional insights.

## 2 Method

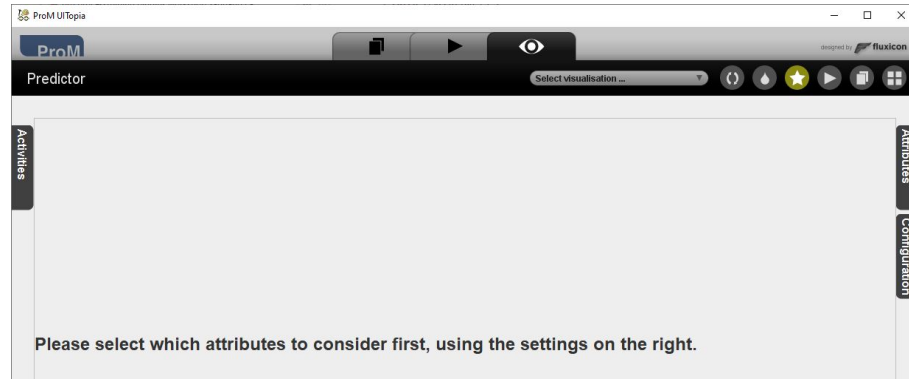
Suppose a problem has been identified in the process and the following information is provided about it.

- (i)  $T$ , the class feature which captures the existence of the problem. The domain of  $T$  is  $\{+, -\}$ .
- (ii) A set of descriptive features  $\{X_1, \dots, X_n\}$ .
- (iii)  $B$ , the sensitive feature where its domain is  $\{\ominus, \odot\}$ .
- (vi)  $\epsilon \in [0, 1]$ , indicating the acceptable level of discrimination against  $B$  (the amount of acceptable dependency between  $T$  and  $B$ ).

Please note that as the attributes can be either a trace or an event level attribute, for the event level attribute, we need to identify not just the attribute names but also the activity names that the attributes belong to.

### 3 How to use the plugin

The inputs of this plugin are the event log, the Petri-net model of the process, and, the conformance checking results of replaying the given log on the given model. The current implementation focuses on three types of problems in a given process: (1) routing problems, (2) deviation problems, and (3) performance problems. The outputs of the plugin are two decision trees, a standard and a fair one. Figure ?? shows the UI of the plugin.



**Fig. 2.** The UI of the discrimination aware decision tree plugin.

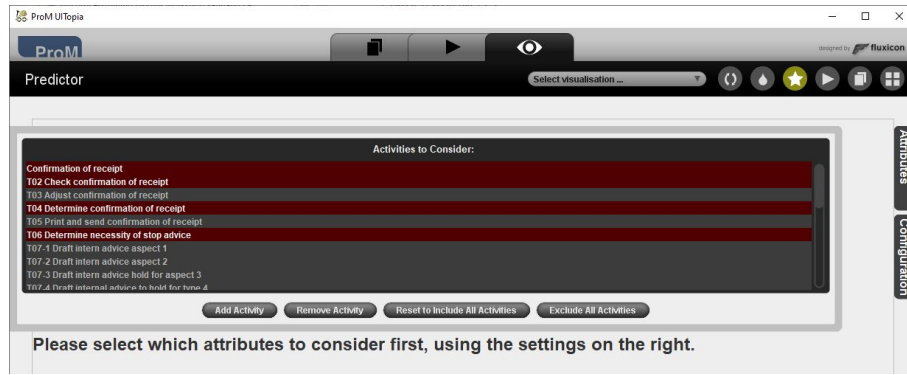
In the following, we explain how the three steps of discovering a fair decision tree is performed using this plugin.

*Enriching the log* An event log includes some trace or event level attributes. However, these are not all the information encoded in an event log. In this step, we add some attribute values to the traces and its events. These added attributes can be related to any one of different process characteristics; time perspective, data flow-perspective, control-flow perspective, conformance perspective, or resource organization perspective. They may be driven from the given log, conformance checking results from replaying the traces on a given Petri-net model, or any external information resource like the weather information.

*Extracting the data* To discover meaningful dependency results by the classifier, we need to capture the data such that the causality relations among them and the class attribute are not violated. To do so, given the class feature  $T$ , we apply the following two rules while extracting the data:

1. If  $T$ , is an event attribute each trace may map to several situations and the data should be extracted from that part of the trace that happens before the occurrence of  $T$ . However, if  $T$ , is a trace level attribute then the data should be extracted from the whole trace.
2. The value of the descriptive feature with the closest occurrence time to the occurrence of  $T$  must be collected.

The second rule is valid assuming that if one of the descriptive features has happened several times before the occurrence of  $T$  in a trace, the one that is chronologically closest to the occurrence of  $T$ , has the most effect on it. The



**Fig. 3.** The *activity* tab to select the activity names that we are interested in their attributes.

setting needed for extracting the data from the enriched event log are as follows:

- If the data includes event level attributes, then we need to select the activity names using the *activity* tab on the left side of the UI (Figure 3).
- Selecting the attribute names in the *attribute* tab on the top right part of the UI (Figure 3). In this tab, the attribute names are categorized in a tree structure. Please note that for some of the attributes more settings are needed. For example, a threshold has to be set for *trace-delay*. As another example, if you select the *choice attribute*, the Petri net model of the process would be demonstrated on a window and you can select some of the choice places on it. The choice made in these choice places will be considered as descriptive features. Similarly, if you select *Sub-model attribute*, you have to select a sub-model of the Petri net model of the process by selecting some of its transitions. The duration of the selected sub-model will be considered as one of the descriptive features.
- Using the *Configuration* tab on the bottom right side of the UI (Figure 3), you can set the class feature.

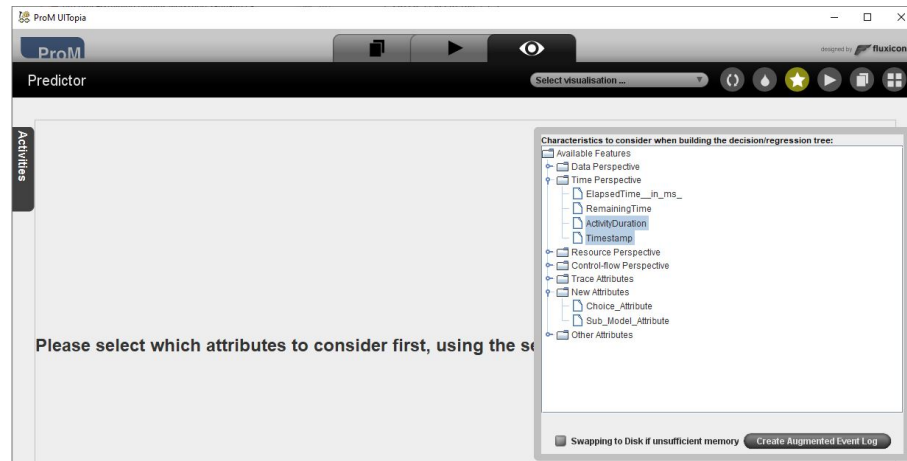


Fig. 4. The *attribute* tab to select the trace and event level attribute names.

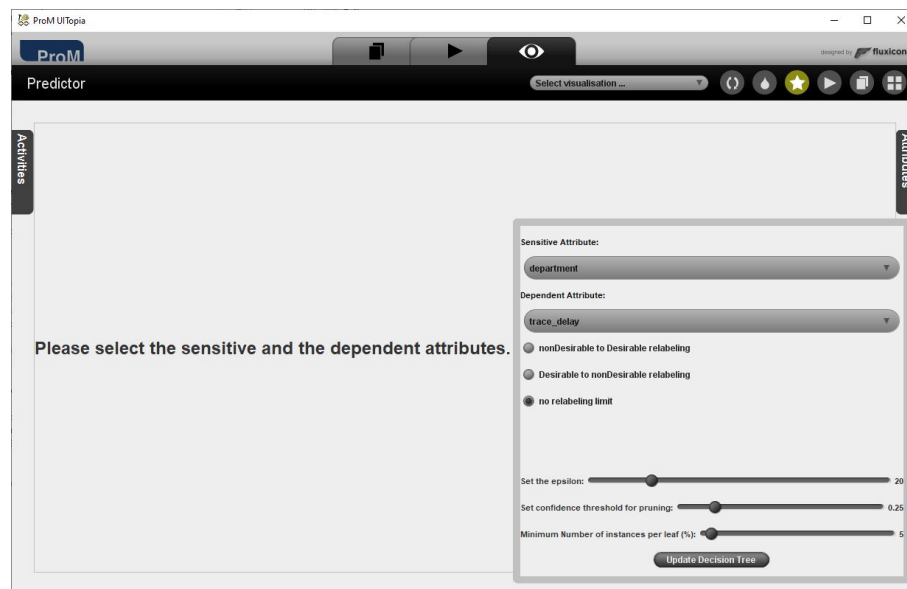


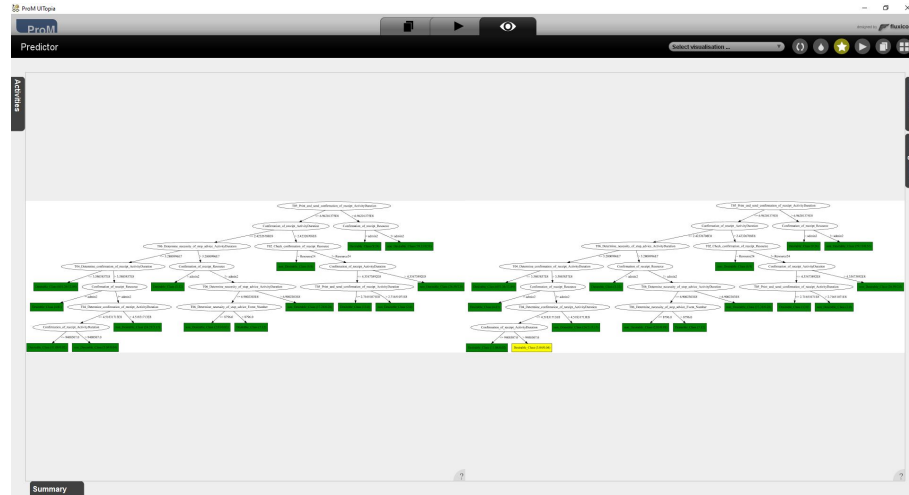
Fig. 5. The *configuration* tab which can be used to set the parameters of fair decision tree.

*Learning fair classifier* To discover a fair classifier (decision tree) the following setting on the *configuration* tab are needed:

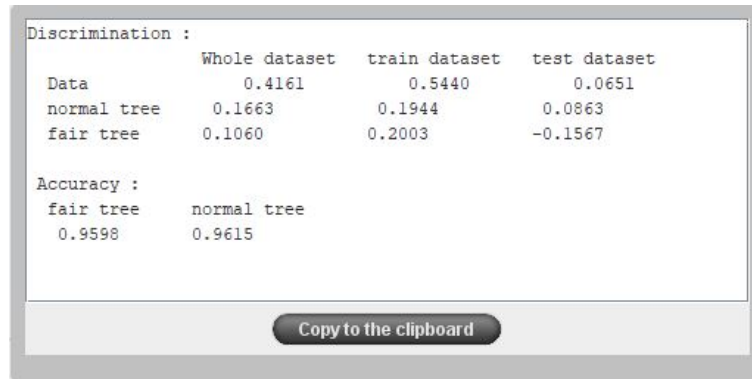
- the sensitive attribute,
- the protected values,
- the class attribute,
- the desirable values of class attribute,
- the type of relabeling,
- epsilon value,
- confidence threshold for pruning, and
- minimum number of instances per leaf.

## 4 Interpreting the outcome

Figure 4 shows an example of the output of this plugin, which includes two decision trees; a standard one (on the left) and a fair one (on the right). More over the level of the discrimination and the accuracy of both decision trees are available using *summary* tab on the bottom of the UI (see Figure 4 for an example). The yellow leaves in the fair tree are those leaves that have been relabeled to decrease the level of discrimination by the classifier. The fair tree, in which the correlation between the sensitive and the class feature is lower than  $\epsilon$ , can be used to make fair root cause analysis for the problem captured by the class feature.



**Fig. 6.** The output of the plugin.



Discrimination :			
	Whole dataset	train dataset	test dataset
Data	0.4161	0.5440	0.0651
normal tree	0.1663	0.1944	0.0863
fair tree	0.1060	0.2003	-0.1567

Accuracy :	
	normal tree
fair tree	0.9598
	0.9615

Copy to the clipboard

**Fig. 7.** The level of discrimination and the accuracy of both standard and fair decision trees are available in *summary* tab.

## References

1. Qafari, M. S., Aalst, W. V. D. (2019, October). Fairness-aware process mining. In OTM Confederated International Conferences "On the Move to Meaningful Internet Systems" (pp. 182-192). Springer, Cham.