ACD2: a BPI 2018 Challenge Use Case

Stephen Pauwels and Toon Calders

University of Antwerp, Antwerp, Belgium
{stephen.pauwels,toon.calders}@uantwerpen.be

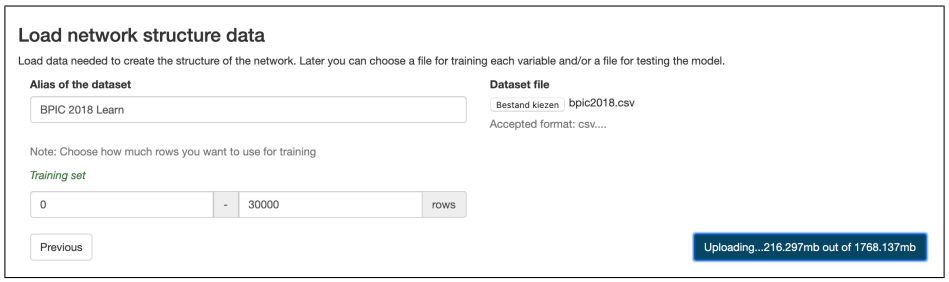
In this document we explain in detail and step-by-step how our tool can be used for analyzing the BPI 2018 Challenge data[1].

The description of the data indicates different documents and procedures that gets used in the different years. During this use case we want to test if our tool can detect these drifts and give an explanation for it.

1 Preparing the data

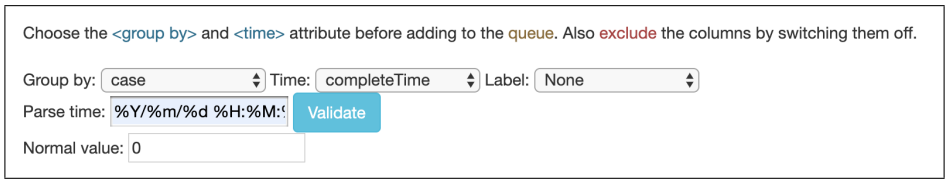
The first step in our analysis is to upload the dataset. At the moment our tool only allows for .csv files. Using Prom we have converted the xes-format to csv.

On the data upload page we can give an alias to our new dataset, we call it 'BPIC 2018 Learn'. Next we select which file from our local machine we want to upload and select the number of rows we want to use for this dataset. For our experiment we set the training set going from row 0 to row 30,000. This is set like this because we want to train our model on the first part of the full dataset and afterwards compare the rest of the data with this training set. Now we can start uploading the file and a first analysis is performed to select the attributes that are present in the data.

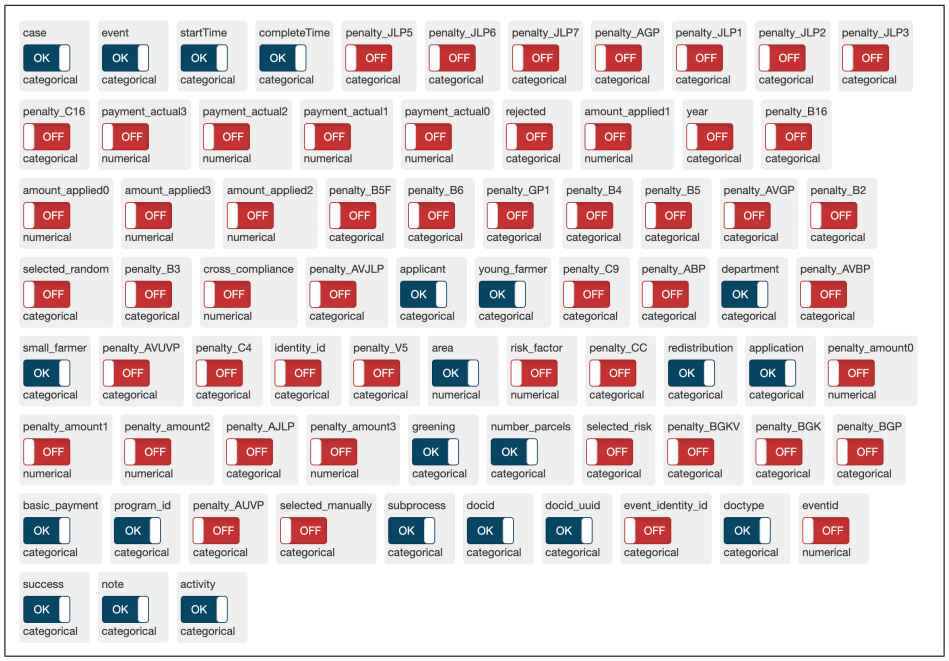


Next we get an overview of all attributes. We also have to indicate some of the special attributes present in the log, like the case ID, the time and optionally a label. We also have to give the exact time format used in the log, so the tool can correctly interpret the time. For the BPIC 2018 data we have to use the following format: *%Y/%m/%d %H:%M:%S.%f*.

[1] https://www.win.tue.nl/bpi/doku.php?id=2018:challenge



Next we select which attributes we want to include in our dataset. The BPIC2018 data consists of two types of data. One type about the application and process itself, which is known when the events are executed. The other type are attributes that indicate the result of the entire process, these attributes we do not want to use in our analysis. We thus only select the appropriate attributes, as shown below. We also remove the year attribute, as it is our goal to detect these by detecting the Concept Drift in the data.

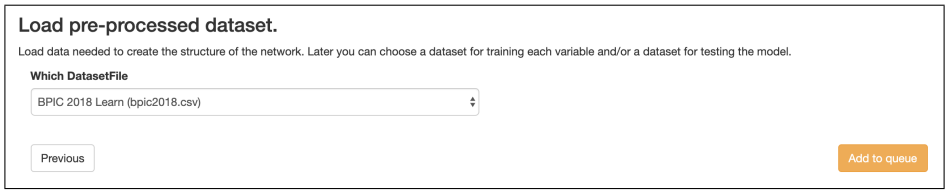


Next we repeat these steps for the testing dataset. Which is identical to the training dataset, but uses all of the rows present in the log.

2 Learning the model

Next we need to learn the model structure for the EDBN that will form the basis of our experiments. First we give some information about the model.

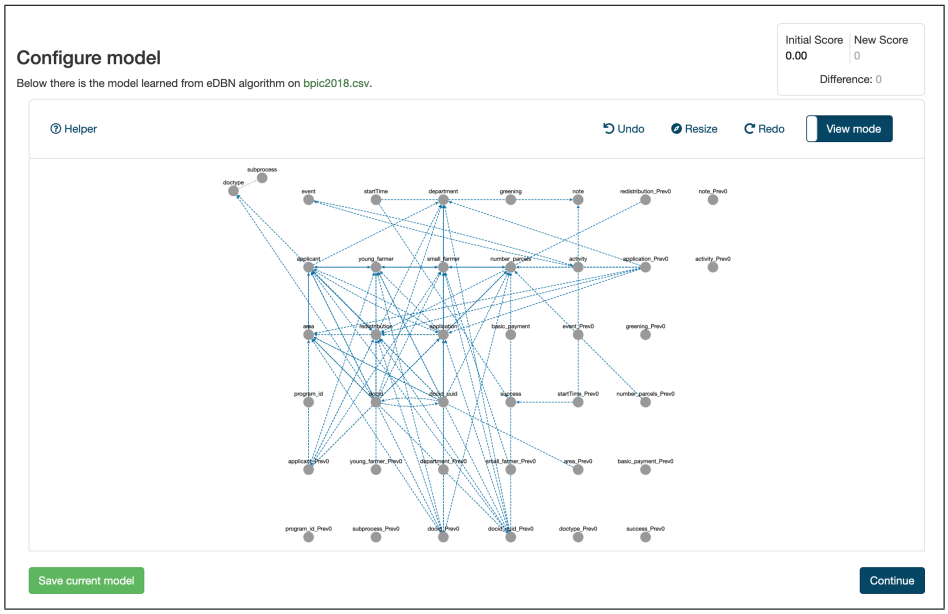
In the next page we select the data we want to use to train the structure of our model. We select the Learn dataset we just uploaded.



The data and request are now sent to the server where the structure learning algorithm now starts. We can see in the *Experiments* tab the unfinished experiments for which he is still computing the model structure.

When the model is learned we see a green indicator instead of a red one.

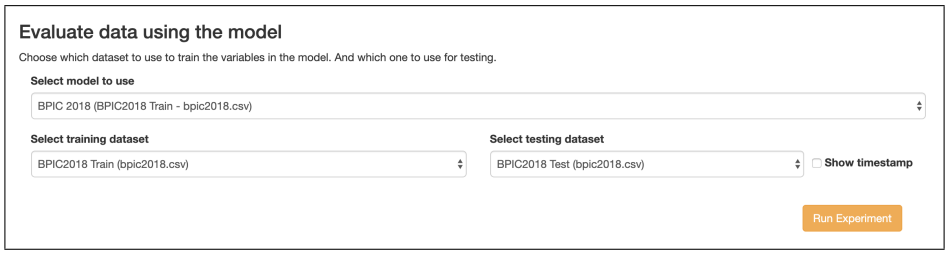
We can inspect the learned model by opening the experiment. We get the network as shown below, if needed we could manually add or remove dependencies.



If we are happy with the structure of the network we can go further to initiate an experiment.

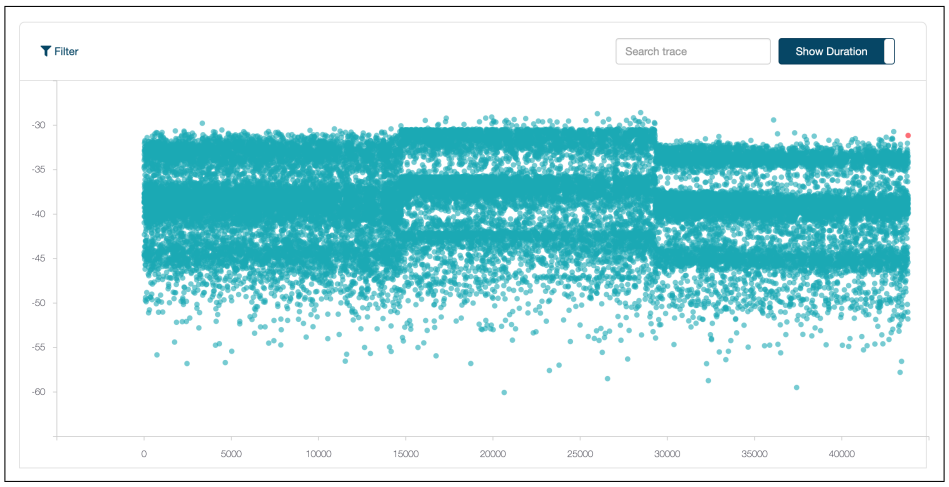
3 Performing an Experiment

We select the file we want to use for training the model and a file for testing the trained model afterwards. We choose the learning and testing version of our datasets we have uploaded. We also deselect the *Show Timestamp* checkbox, as it is our goal to find the differences between the years using the attributes that contain no direct timing information. When everything is entered correctly we can continue.

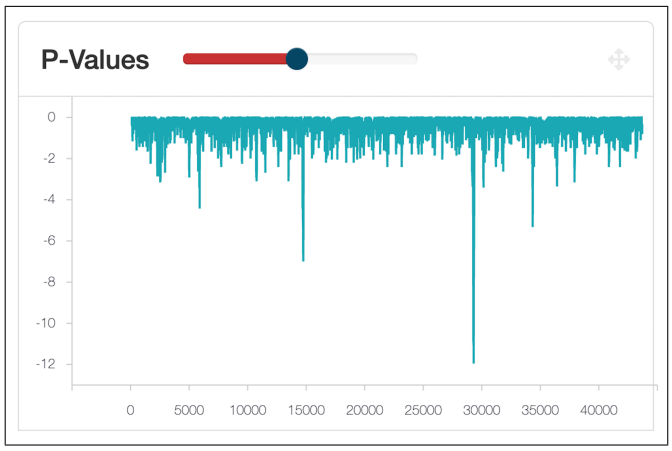


When the model has finished training and all events are scored we can take start analyzing the results.

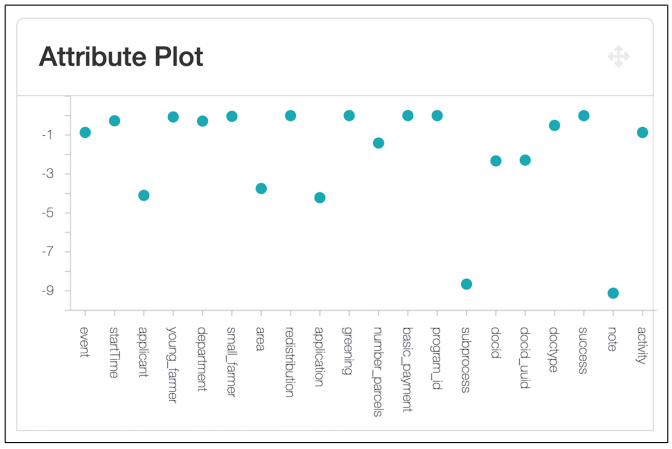
The first step in our analysis is to take a closer look at the trace graph. In this graph we can look at possible clusters or outlying traces. Below we get a detailed view of the graph.



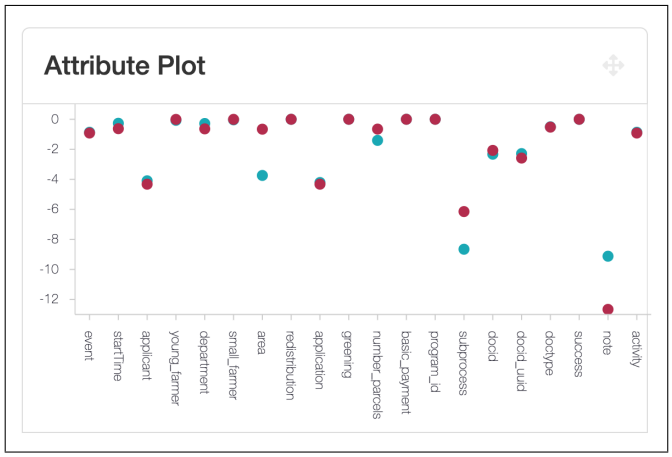
Visually we can already distinguish three regions with a clear difference in the distributions of the scores. When looking at the P-values graph we can confirm our suspicions.



Next we can start investigating what attributes do cause these differences between the regions. In order to do so we use the Attribute Plot. Below we can see the mean score for every attribute over the entire test dataset.



After applying a filter based on the found drift points we get the following graph:



Which is in the line with our initial findings presented at the BPI 2018 Challenge. All used datasets are available in the demo tool which can be found online at http://adrem.uantwerpen.be/conceptdrift.