DeviationSequenceAnomalyDetection operator

This operator searches for anomalies in a sequence in comparison to the learned sequences. To do this, the operator uses the deviation information created by the DeviationSequenceLearn operator. For each tuple of a sequence, the value is compared to the learned distribution (mean and standard deviation). The data input port is 0, the input port with learn data is 1.

Parameters

- **interval** Defines, how many standard deviations are allowed for a tuple to be different from the mean. 3.0 is the default value. Choose a smaller value to get more anomalies
- **tupleCountLearnAttribute** The attribute name on the learn port that gives the group count (the counter that gives each tuple in the sequence a number)
- **meanLearnAttribute** The attribute name on the learn port that has the mean
- **standardDeviationLearnAttribute** The attribute name on the learn port that has the standard deviation
- **valueDataAttribute** Name of the attribute which should be analysed
- **GROUP_BY** If you use a deviationSequenceLearn operator, use 'group' as grouping attribute.
- **fastGrouping** Use hash code instead of tuple compare to create group. Potentially unsafe!

Example

The DeviationSequenceAnomalyDetection operator uses the learned data from the previous DeviationSequenceLearn operator.

```
#PARSER PQL
#RUNQUERY
///
 Values above 50 will be 'true' (which means that the current sequence
starts / runs) and smaller values to 'false' (means: sequence ended)
stateInfo = MAP({
                expressions = ['temp', ['temp > 50', 'state']]
              },
              System.manual
            )

/// The elements within one sequence will be counted (starts from 1 with each new sequence)
sequence = MAP({
              expressions = ['temp','counter(state)']
            },
            stateInfo
          )

///
 The tuple which marks the end of the sequence (and itself is not part
of the sequence) has the counter_state_ 0 and will be filtered out
onlySequence = SELECT({PREDICATE = 'counter_state_ > 0'}, sequence)

/// Learn how a "normal" sequence is. The first 15 sequences will be learned and used as the definition of
"normal"
sequenceLearn = DEVIATIONSEQUENCELEARN({
                  group_by = ['counter_state_'],
                  parameterAttribute = 'temp',
                  sequencesToLearn = 15
                },
                onlySequence
              )

///
 Check, if the current tuple of this sequence differes from the normal
tuples of the sequence at the specific point of the sequence
sequenceAnalysis = DEVIATIONSEQUENCEANOMALYDETECTION({
                    interval = 4.0,
                    standardDeviationLearnAttribute = 'standardDeviation',
                    group_by = ['group'],
                    meanLearnAttribute = 'mean',
                    valueDataAttribute = 'temp'
                  },
                  0:sequenceLearn,
                  1:sequenceLearn
                )
```