Assignment 2 CDA

Tim van Rossum, 4246306
Michiel Doesburg, 4343875

May 26, 2018

1 Familiarization with the data

The dataset contains a total of 44 signals, with some being static values, other being sinusoid signals, and others being partially discrete (as in, they can rise or drop at certain points). L-signals tend to be more sinusoidal, F-signals tend to be "partially discrete", and S-signals are all binary signals.

All the signals together seem to have little correlation. Some combinations of signals are reasonably correlated: e.g. L_T3 and P_J302 have a 0.42 correlation coefficient. Moreover, the signal P_J302 in figure 1 looks cyclic. A cyclic signal like this is much easier to predict than a 'random' signal, since there is a rough base model that it adheres to. This signal's values are within a clear band at about 2-6 on the Y-axis. In such a case it is very easy to come up with a basic anomaly detection technique: if the signal moves outside this band this would be a clear sign of an anomaly. Plotting the mean error we saw for the ARMA prediction on signal L_T1, we saw that there were only 2 cases were the error was greater than three times the standard deviation.

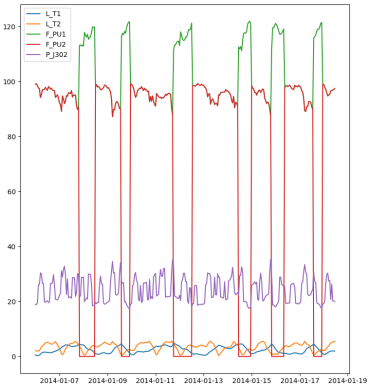

Figure 1: Visualization of some signals. The red and greens signals are partially discrete.

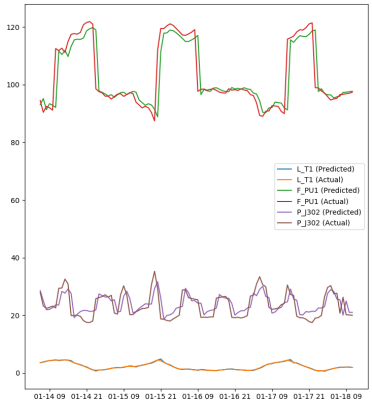

Figure 2: ARMA predictions on some signals. The predictions work better on signals.with less variance

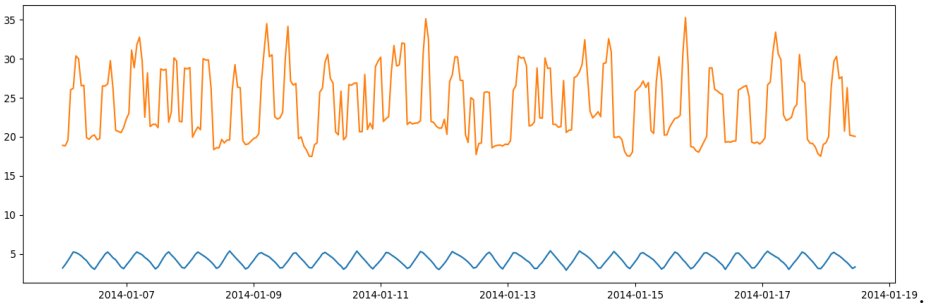

Figure 3: Signals L_T3 and P_J302. The peaks are roughly aligned.

2 ARMA

The script `ARMA.py` learns a ARMA model for any particular sensor. The parameters for the model are determined by testing out different p's and q's and then looking at the AIC value of the fitted model. A lower AIC means a better fit. Fitting the models takes quite some time, and as such the search space for parameters is limited. For p, only values in the range of $[0, 8]$ are tested, and for q, only values in the range of $[0, 2]$ are tested. Both these ranges are inclusive. We thought these ranges would still allow for a reasonable search space while limiting the time taken to search for the ideal parameters.

After the predictions are done, the predicted values are compared to the actual values. The difference between the two is used, and then the mean of the absolute values is computed, along with the standard deviation of the absolute values. An attack is detected if a predicted value differs from the actual value by more than the absolute mean plus three times the standard deviation. The sensors that have cyclic time series can be modelled effectively by ARMA, and the anomalies that can be detected are contextual anomalies.

3 Discrete models

For this task, we used the SAX method as shown in class. The script `SAX.py` is a script that can discretize a time series using SAX. The script was provided by Qin Lin, and we altered a few things to make it work with Python 3. Discretization of the signal in this way creates a representative string of characters of the signal, which allows us to perform sequential data mining methods on it. This is in contrast to PAA, which only discretizes the signal, which does not allow us to perform these data mining methods. Discretization by use of SAX makes sense as the signal values are now grouped by how much they differ from the mean of the signal.

The script `applying_SAX.py` does not only apply SAX on the signals, but also applies the N-grams data mining method to find anomalies. An N-gram is anomalous (and possibly an indicator of an attack) if the score is lower than either the mean score minus three times the standard deviation, or 0.3, whichever is higher. The 0.3 was chosen to prevent the score boundary from becoming negative. The sensors that do not have cyclic time series can be modelled effectively by SAX, and the anomalies that can be detected are collective anomalies.

4 PCA

5 Comparison