Delft University of Technology

CYBER DATA ANALYTICS
CS4035

ANOMALY DETECTION

Authors:

Arpita Ravindranath(5002702), Menno Bezema (4248252)

May 26, 2020

1 FAMILIARIZATION TASK

1.1 What types of signals are there?

We use pandas DataFrame:describe() method to get insight into the data contained inside the dataframe. There are 44 columns in total, 16 of which being integers and the rest floating point values. The integer values are all either 0 or 1, and nothing else.Thus, we can say there are 2 main types of data, boolean values and numerical, these belong to the actuators and sensors respectively.The website https://batadal.net states: "the flow data unit is LPS, pressure and water level units are meters." This suggests that the F_* columns contain flow data, and the L_T* columns describe water levels.We have the columns L_T1, ..., L_T7. All of them are floating point values between 0 and 5.5, likely they describe the same kind of signal, presumably water levels in different tanks.

Then we have the F_PU1, ..., F_PU11 and S_PU1,..., S_PU11. The F-columns are floating point values and the S-columns booleans. The naming suggests these are linked, and thus likely describe an actuator linked to a sensor in some way. Some F_PU columns contain all zeroes, its worth noting that if this is the case, the corresponding S-column contains all zeroes as well.

We have columns F_V2 and S_V2 containing similar data as the colums described above, S_V2 being boolean and F_V2 float.Then come another 12 float columns named P_J followed by a (seemingly random) number they seem to contain similar data as the F_PU columns.Lastly there is ATT_FLAG, another boolean column indicating the presence of an attack or not. We plotted the graph of the values to show when the attacks occur.

1.2 Are the signals correlated? Do they show cyclic behavior?

If we make a heatmap we also see that some columns correlate heavily, also between the P_J and F_PU columns. For example, there is a strong negative correlation between F_PU1 and P_J269 and between F_PU2 and P_J280. There is a strong positive correlation between F_PU4 and P_J256 (among many others).

If we take a few of the signals and plot them together, we see that many of them are cyclical, at least the F_PU* signals and L_T* move in regular intervals. The P_J* signals seem to be less regular, although they still move up and down with a fairly steady period and amplitude. Additionaly, at least some of them are definitely correlating signals, like F_PU1 and F_PU2.

1.3 Is predicting the next value in a series easy or hard? Use any method from class.

A Moving Average predictor was used to predict the series. A prediction is made by taking the average value of the last few points in the dataset as the prediction of the next value. For a window size of x the first x points are discarded, then point x + 1 is predicted by taking the mean of the x points. Then, point x+2 is predicted by taking the mean of x + 1, x and x - 1 (The actual value point x+1, not the predicted one from before).

Using this we predict the values of 3 randomly selected sensors: F_PU1, P_J14, L_T1. The resulting mean square error for windows sizes 1 upto 4 is printed in the code. For F_PU1 and P_J14 the predictions are not that accurate. For L_T1 the predictions are very accurate as can be seen by the low mean squared error, we also show a plot of the predictions so you can see how close they get.

Finally we conclude that the usage of small windows is better than that of larger windows. This might come in handy when performing n-grams.

2 LOF task – Individual(Arpita)

2.1 Plot LOF scores

In order to do LOF we first need to do some pre processing. In order for LOF to work correctly the data needs to be normalized such that each feature will be weighted equally in our calculations.

2.1.1 Plot the LOF Scores on training data 1 as a signal for several numbers of neighbors

We then proceed to answer question 1.Here we Train LOF on dataset 1 and setting the novelty to true as the training data contains no anomalies.We then plot the value of the no of errors and the no of correct detections for different neighbor values.The default distance metric - minkowski is used

2.1.2 Select a number to use and justify this choice using the obtained LOF scores and detected anomalies.

Next we need to tune the parameters, primarily the number of neighbours we will use.We want the number of neighbours that generates the least amount of errors and at the same time generates a good number of true attack detections.Based on the plots we choose number of neighbors to be 5.

2.2 Analysis and answers to the questions

2.2.1 Do you see large abnormalities in the training data? Can you explain why these occur? It is best to remove such abnormalities from the training data since you only want to model normal behavior.

We see from the plot that there are some datapoints showing a drastically different LOF score these abnormalities may be due to degradation,unexpected failures or human errors etc.Since they do not contribute to the "normal working" of the components these data points are dropped.We drop all the data points which have an negative LOF score of lesser than -1.3 based on the above plot.

2.2.2 Describe the kind of anomalies you can detect using LOF

LOFs detect those datapoints that belong to a lesser dense region when compared to its neighbors.In the above cases we make a comparison between 5 neighbors.Comparing the plots before and after remove abnormalities from the training set we see that the number of true positives has increased but this has also resulted in an increase in the number of FP.

3 PCA task - Individual(Menno)

In order to do PCA analysis we first need to do some pre processing. In order for PCA to work correctly the data needs to be normalized so that it has a mean of zero and unit-variance, such that each feature will be weighted equally in our calculations.

3.1 Plot PCA residuals

3.1.1 Plot the PCA residuals for different number of components on training data 1 in one signal.

We did not understand what was meant by different number of components as it is not possible to fit multiple components on 1 signal. The second stap in PCA is to filter out abnormalities in the training dataset. In order to find any abnormalities if they exist we plot the (squared) PCA residuals. This plot is shown in the residuals plot generated by the cell below.

3.1.2 Do you see large abnormalities in the training data? Can you explain why these occur? It is best to remove such abnormalities from the training data since you only want to model normal behavior:

This plot shows a few clear abnormalities. We think they occur when there is either a bad reading or there is an attack. These need to be removed as they can affect the principal components. We do that in following cell: Next we need to tune the parameters, primarily the number of principal components we will use. We need a number of components that captures a big percentage of the variance. In order to decide this we plot the fraction of variance captured by each principle component, for this plot see figure varplot.

Then, we plot the cumulative variance captured by an increasing number of principal components in figure cumulative variance plot. Reading from this plot we can see that at around 12 principal components, about 99% of the variance should be captured. The thresholds are set to the maximum and minimum residual of the training data, (this will result in 0 false positives in the training data).

3.2 Analysis and answers to the questions

3.2.1 Describe the kind of anomalies you can detect using PCA.

We now run the PCA analysis using 12 principal components on the test data. Any resulting residuals that are either above the maximum threshold or below the minimum found using the training data are marked as possible attacks. The PCA analysis results in 93 true positives being detected and 5 false positive, see the graph below for a visual representation. It detects all attacks and some extra. If we increase the maximum threshold by a factor of 2, and decrease the minimum by dividing it by 2, we can change the model so that it generates 52 true positives and only 1 false positives. Using these exaggerated thresholds we decrease the false positives by a lot. In the end It might be useful to use both and definately investigate all attacks with exagerated thresholds. And check on the remaining attacks detected by normal parameters.

4 ARMA task – Individual(Arpita)

First we ensure that we split DATETIME to year,month,day and hour and then perform normalization to ensure that all features are weighted equally.We select the sensors to be used by looking at the correlation matrix and removing highly correlated signals.Then the signals chosen are those that are affected by the attacks mentioned on the website(except for P_J300).The following subsections cover each of these signals - F_PU7,L_T4,L_T1,L_T7,F_PU10,P_J300.Here we cover the details for signal F_PU7

4.1 Print relevant plots and/or metrics to determine the parameters.

4.1.1 Use autocorrelation plots in order to identify the order of the ARMA models.

We calculate the autocorrelation and partial autocorrelation functions to make an informed descision about what ARMA parameters to use.

4.1.2 The parameters can be determined using Akaike's Information Criterion (AIC) or another model selection method. Note that there exists a wide range of ARMA variants; you only have to use the basic model.

We use auto-correlation (ACF) and partial auto-correlation (PACF) plots as a guide to find the optimal values for the order of the fitted model.Regarding the Autcorrelation plot we can see that for higher values after around 6 lags autocorrelation drops into the blue zone which represents the confidence interval.With respect to the PACF plot there is significant drop after around 6 lags if we look into the higher values.The AR term is estimated from the PACF plots while the MA term is estimated from the ACF plots.

The function find_arma_order implements the grid search and returns the model with the lowest AIC. In the current case that is ARMA(4,5)

4.2 Plots to study the detected anomalies

4.2.1 Plots the residual errors and study some of the detected anomalies.

The threshold for detecting anomalies is extracted based on the standard deviation of the residuals extracted from the training signal. We set threshold as 2 * std_deviation and when residuals in the test set are greater than this threshold they get classfied as anomalies.This value is selected in order to produce a good accuracy value and create a balance between detected FPs and TPs. From the performance metrics it is seen that this model doesnt detect very many attacks with having low values of TP and FP The same procedure is repeated with the following signals.

4.3 Analysis and answers to the questions

We have plotted the graph for the residuals along with the actual and detected attacks.We have zoomed into the graph to show attacks that have directly affected the sensor.Based on these graphs we notice that the attacks detected are those attacks that directly affect the sensor on which we run ARMA.But often a lot of false postives are also generated.ARMA detects sudden anomalies and not attacks lasting for a long period.The reason for this is previous datapoints are used in ARMA to make predictions.

5 N-gram task – Individual(Menno)

To use discrete models we first need to discretize the data. This means that we convert the continuous data from the dataset into discrete data like for example high medium and low values. To do this we use Symbolic Aggregate Approximation (SAX). We picked this method as it is easy and fast to implement, it offers the ability to select the amount of segments you want to divide the dataset into as well as the amount of symbols to use to describe the final dataset. An anomaly is flagged if the n-gram of a datapoint is not present in the training set.

For distance we look at the occurance of a specific n-gram, if it is present in the training set then it is not anomaly. If it is not present in the training set however it is classified as an anomaly. As you can see in the figure below, the dataset is converted into a discrete dataset with in this case 2 symbols. The image shows a significant reduction in dimensionality. In our implementation we convert the numbers given by sax to letters. The table is shown in the next cell: Below we tested every single sensor. The sensors with decent results are displayed in the table below:

From the above we can say that F_PU6, S_PU6, F_PU7 and S_PU7 can be modelled effectively using n-grams

It looks like S_PU6 is a really good predictor so we plotted its results 2 cells below. It finds its true positives in just 2 attacks. Combined with 1 false positive. This is useful because if N-grams designate an attack as being an attack it most certainly is an actual attack.

For F_PU7, also plotted below, it correctly classifies a single attack and also gives a false positive.

sidenote: A better method for finding a globally optimum set of parameters might be able to find other results. Running this yourself takes a long time (50 minutes) so you can either trust the output or run it again :) Or verify the output by running for a single signal.

6 Comparison Task

We compare the algorithms on the basis of a couple of factors:

- True and False positives, each positive means someone needs to look at an alert and possibly take action so we want little false positives and many true positives.
- Number of attacks undetected, if an algorithm has 1000 true positives all within 1 attack it might look like a good algorithm but it still misses all the other attacks.
- Time of detection, if an algorithm detects an attack immediately that is better than when it detects it later. Of couse no detection at all is even worse.

The LOF analysis

- Basic: 56 true positives, 37 false positive.Removing abnormalities: 253true positives, 123 false positive.
- Both detects all the attacks
- Both methods detect attacks pretty fast.

The LOF analysis shows us that along with a large number of true positives that are detected there are also quite a large number of false positives.So while it is useful in detecting attacks it generates a lot of overhead due to the unnecessary effort that goes into looking at the many false positives.Based on the plots we get before removing abnormalities from the training set, we notice that the LOF analysis helps in detecting the end of an attack.This is not very useful as we would prefer an earlier detection so that the attack can be stopped and recovery measures taken.

The PCA analysis is very promising

- Basic: 93 true positives, 5 false positive. Changed parameters: 52 true positives, 1 false positive.
- Basic: detects all attacks. Changed parameters: misses 1 attack.
- Both methods detect attacks pretty fast.

In conclusion PCA is a very good method to use for anomaly detection. Based on the preference of the analyst responding to the attacks you can chose for basic (detect all attacks and some false positives) or changed parameters (detect almost all attacks but very little false positives)

The ARMA analysis

- As can be seen from the plots in subsection 4 the number of generated false positives are really high compared to the true positives. So using an ARMA methods for detecting attacks again is not very suitable for use practically.
- It does detect a lot of different attacks but combines this with false positives.
- Contrary to LOF ARMA helps in detecting the start of an attack. It detects sudden deviation from normal behaviour and hence can be used to detect short attacks. This technique detects an attack when there is a clear deviation in the residual pattern

Signal	TP	FP
L_T1	28	32
L_T4	0	12
L_T7	2	22
F_PU7	0	1
F_PU10	2	0
P_J300	10	86

The N-gram analysis

1. Ngrams show a lot of true positives and little false positives, this is advantageous because if an alert goes off you can respond to it knowing it is a high chance of an actual alert.
2. Ngrams score badly on this criteria because it detects a just 2 attacks, but detects them well.
3. Ngrams do however detect attacks pretty quickly.

In the end ngrams are useful because if they give an alert you know you need to respond to it.

Based on all the above, PCA seems the best method to implement. If you want to detect all attacks and dont mind a tiny amount of false positives then it is very useful. One could even think of combining the different PCA methods to gain insight into what are actual attacks and what attacks might need some further investigation.