Team 8 | Rayson Lim (1002026) | Kimberlyn Loh (1002221)

The Analytics Edge (Fall 2018) – Data competition

1 Approach Developed

1.1 Elementary Data Analysis

From the training datasets, we can first make several observations:

1. There are several different types of attacks that have different telltale signs
2. The first training dataset has several anomalies
3. The second training dataset has no anomalies if we filter out observations with attacks
4. Attacks happen over a duration and hence, observations cannot be treated independently
5. Regions in the second training dataset under attack can have (1) changes in averages, (2) changes in amount of fluctuations, (3) changes in number of observations above the mean value or any combination of the above signs
6. We are dealing with time series data that comes with a lot of noise and fluctuations

The above 6 observations form the basis of our model.

1.2 New Features

First, we create new features to detect signs of an attack. We calculate moving averages for every numerical feature. We also calculate fluctuations which we define as sum of absolute differences between each row and its previous row in a window. This number increases if the value of a feature for a row differs greatly from its neighboring rows. Finally, we define a feature "above" as the number of observations above the mean within a window. This allows us to detect anomalies where the values in a region are not too far away from the mean but are held above or below the mean longer than usual. The definitions for our new features are as follows:

$$Moving\ Average = \frac{1}{|S|} \sum_{x \in S} x$$

$$Fluctuations = \sum_{i=k}^{k+|S|} |x_i - x_{i-1}|$$

$$Above = count(obv. > mean)\ \forall obs \in S$$

where S is the set of x values in the window and k is the row number of the first x value in the window. For any row i, we use the last |S| rows before it to calculate the new features. The window size |S| is a hyperparameter.

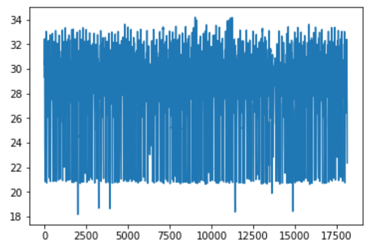
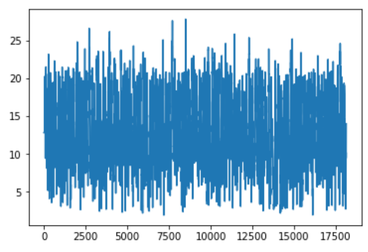
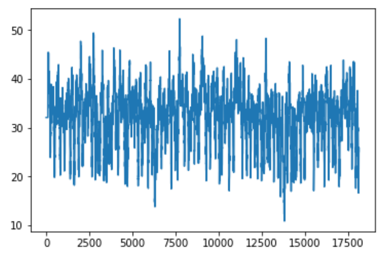
Original	Window size of 40	Window size of 100
		

Figure 1. Plots of Fluctuations for PRESSURE_J300 with different window sizes

As seen from figure 1, as we increase the window size, our averages and fluctuations start to become more representative of the entire dataset. At some point, it would be too general for us to use to predict the ATT_FLAG for a specific region. |S| should be at most the length of the shortest attack in the training set, which has a length of 96 observations. After experimenting with window sizes ranging from 10 to 70, we found that window size of 40 gives us the best F1 score when we use our algorithm to predict ATT_FLAG for the second training dataset.

1.3 Noise Reduction

However, even with newly added features, there is still plenty of noise observed in all the features. In order to counter this problem, a low pass filter is used to make anomalies more obvious. Through experimenting with low pass filters of different sample rates and cut-off frequencies on the second training dataset, we visually saw that a low pass filter with a sample rate of 10 Hz and a cut-off frequency of 3 Hz did a good job at amplifying anomalies.

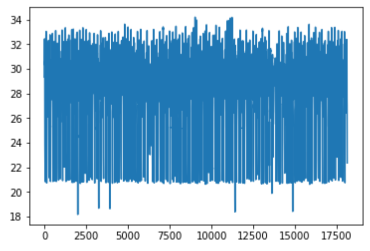
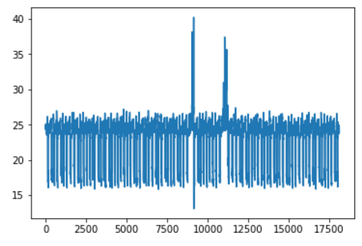
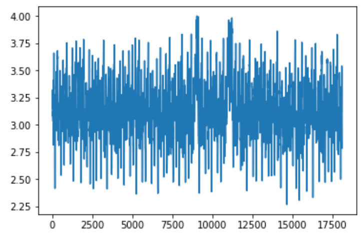
Original	Sample Rate of 10 Hz	Sample Rate of 50 Hz
		

Figure 2. Plots of PRESSURE_J300 with different low pass filters applied to it

1.4 Model

Based on our elementary data analysis, we noted that our model has to be independent of the type of attack that is being performed on the water distribution system as the types of attacks in the test data set may not follow the same patterns as those in the training dataset. Hence, we decided to (1)detect anomalies in the data and flag them as attacks. This seems like a reasonable approach as the second training dataset has no anomalies if we only look at observations with ATT_FLAG set to false. Since the first training dataset has anomalies, we will not use it for training. This almost means that (2)we cannot remove features that were not indicative of attacks in the training data from the model as these features might be indicative of other types of attacks not present in the training set. Finally, since attacks happen in a duration, (3)observations cannot be treated independently.

Combining these points, we came up with the following algorithm:

1. Calculate the means and standard deviations of the numerical features in the second training dataset for the observations that were not under attack
2. Calculate the p-values of each numerical feature for new observations by making normal distribution assumptions
3. Count the number of p-values that are under a predefined threshold, p_{min}
4. Calculate the score of each observation using the following formula

   $$score(1) = votes(1)$$

   $$score(i) = \propto\ score(i-1) + votes(i), 0 < \propto < 1$$

   where i in the row number, α is the decay rate and votes(i) is the number of p-values lower than the threshold. Now every score takes into account all the scores before it.
5. For each observation, if its score is higher than a predefined threshold $score_{min}$, we say that that observation is under attack.

Our algorithm has several hyperparameters, the minimum p-value threshold p_{min}, the decay rate α and the minimum score threshold $score_{min}$. To obtain these values, we perform a grid search using the second training dataset set by maximizing the F1 score of our predictions on the training data. After several hours of searching, we managed to achieve a F1 score of 0.802 on the training data using p_{min}= 0.0073, α= 0.943 and $score_{min}$= 108.

2 Results

As shown in the table below, our model performs well on both the training and testing data. In addition, using a low-pass filter is proven to be effective as the model improved significantly after we add the low-pass filter. Moreover, we can see that a normal decision tree model severely over fits to the training data from how different the F1 scores are between the training and testing data sets. This is because the decision tree model cannot predict attacks that follow patterns that the model has never seen before. This is not so apparently in our custom model.

Model	Training F1 Score	Kaggle Public F1 Score
CART (MA, Fluctuations)	0.94046	0.46932
Custom model using votes (MA, Fluctuations)	0.71137	0.69473
Custom model using scores (MA, Fluctuations, Above)	0.78135	0.71411
Custom model using scores (MA, Fluctuations, Above, Low-pass Filter)	0.80218	0.75545

Table 1. F1 scores on training and test data for different models

3 Interpretability

Our model is fairly simple to interpret.

A very low p_{min} suggests that the values of our features change a lot during a cyber attack. A α value close to 1 suggests that if the previous observations are under attack, the current observation is many times more likely to be under attack. This agrees with our original observation that attacks happen over a duration of time.

To determine which tanks and pumps are affected by an attack, we can simply plot the p-values we calculated in step 2 of our algorithm. For instance, if tank 1 is affected, we will see a low p-value for the feature "LEVEL_T1". In addition, we can see how it is affected by plotting the p-values of the moving average, fluctuations and above counts. If the p-value of the moving average is low, we can conclude that the particular feature is being shifted away from its mean. If the fluctuations' p-value is low, we can conclude that the particular feature is fluctuating a lot more or a lot less than usual. Finally, if the p-value of the above counts is small, we can conclude that the particular feature is held below or above the mean longer than usual.

4 Limitations of Approach

The most obvious limitation of our approach is that we are detecting anomalies and classifying all of them as attacks. As evident from the first training data, not all anomalies are attacks. However, it is difficult to address this problem as attacks can have different patterns and can evolve over time. Given more data, we may be able to distinguish between true attacks and non-attacks among anomalies.

Secondly, our model ignores the categorical features provided to us. During our elementary data analysis, we did observe that some of the distributions of the categorical features do vary during attacks. However, when we tried to incorporate them into our model, our F1 score on the training data fell significantly. We had yet to think of a better way to utilize them.

Thirdly, our model relies heavily on previous observations in order to detect an attack. The new features we have created all require a window of observations to calculate and our scoring method requires all previous scores. This means that we can only detect an attack a few hours after it has started. This could be a huge disadvantage if we need to detect attacks in the shortest time possible.

Finally, our model assumes that all numerical features follow a normal distribution. This is not a valid assumption as plots of several features show that they are skewed. This affects how well we classify data points as anomalies using the p-value.