Pattern Recognition with Air Quality Sensor Data

Zhou (Joe) Xu
06/19/2020

1. Introduction

This project is part of my freelance data science work for a client. There is no non-disclosure agreement required and the project does not contain any sensitive information. So, I decide to showcase the data analysis and modeling sections of the project as part of my personal data science portfolio. The client's information has been anonymized.

In the project, two datasets are provided, each consists of one week of air quality sensor readings. They are provided to accomplish the following four tasks:

1. Visualize inter-dependencies of the features in the dataset
2. Find anomalies in the data set to automatically flag events
3. Categorize anomalies as "System fault" or "external event"
4. Provide any other useful conclusions from pattern in the data set

In this report I am going to briefly walk through the steps I use for data analysis, visualization of feature correlation, machine learning techniques to automatically flag "system faults" and "external events" and my findings from the data.

2. Exploratory Data Analysis

The datasets are first imported and concatenated into one Pandas dataframe in Python. Some rearrangements are made to remove columns except the 11 features that we are interested in: Ozone, Hydrogen Sulfide, Total VOCs, Carbon Dioxide, PM 1, PM 2.5, PM 10, Temperature (Internal & External) and Humidity (Internal & External).

The timestamps span from May 26 to June 9, 2020 (14 whole days in total) in EDT (GMT-4) time zone. By subtraction, different intervals are found between each reading, ranging from 7 seconds to 3552 seconds. The top 5 frequent time intervals are listed below in Table 1, where most of them are close to 59 and 60 seconds, so it can be concluded that the sensor reads every minute. However, the inconsistency of reading intervals might be worth looking into if it is no deliberate interference involved since it might cause troubles in future time series analysis.

	Interval (s)	Counts
0	59	9279
1	60	8243
2	61	314
3	58	211
4	38	145

Table 1: Top 5 Time Intervals of the Sensor Measurements

For each feature, the time series data come with different scales, so normalization is applied in order for better visualization and machine learning efficiencies. They are then plotted and visually inspected to discover any interesting patterns.

2.1 Pattern Changes

Some of the features seem to share similar pattern changes at specific time points. Three of the most significant ones (Temperature External, Humidity External and Ozone) are shown below in Figure 1. It can be clearly seen that the areas highlighted with pink tend to have flat signals while the unhighlighted areas are sinusoidal.

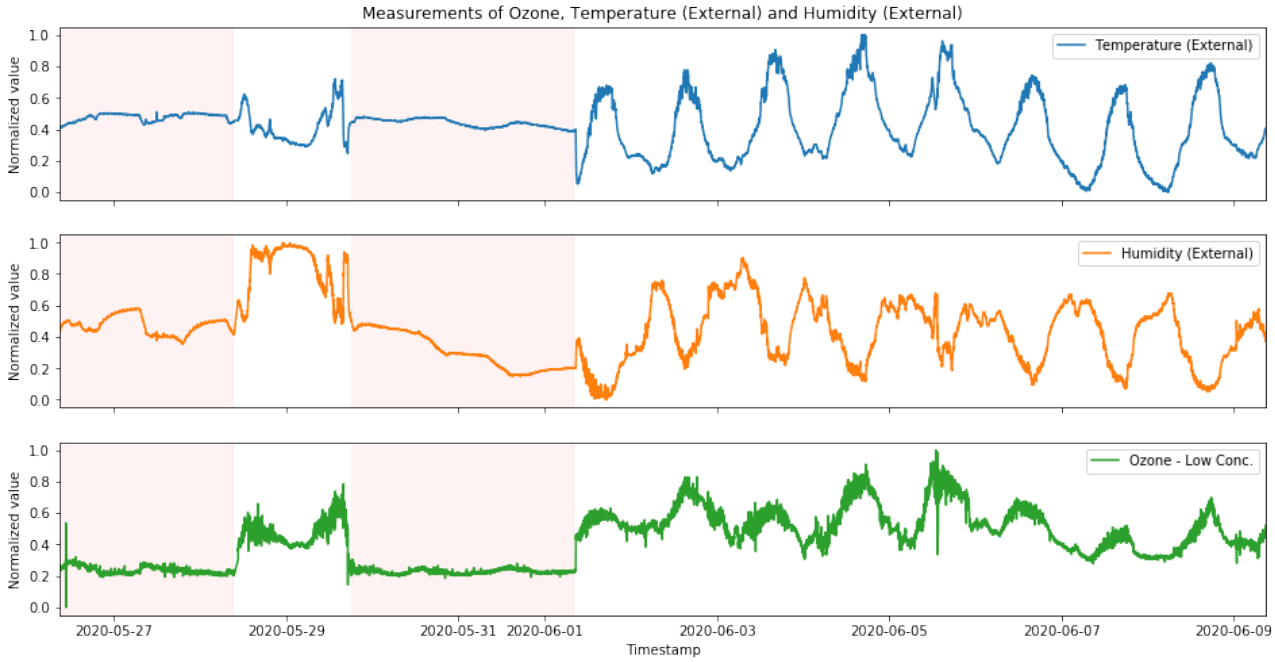

Figure 1: Readings of Temperature (External), Humidity (External) and Ozone. They experience similar pattern changes at same time points.

According to common sense, the outdoor temperature reaches its high point at noon and goes down at night, and this might infer that different test environments were involved during this 14-day period. To test the idea, Toronto weather data is queried from Canada Weather Stats [1]. The temperature and relative humidity are overlaid and compared with the external temperature and humidity in this dataset. The plot is shown in Figure 2. It can be seen that the actual temperature and humidity fluctuate in a sinusoidal fashion. Most parts of the temperature and humidity readings correlate well with the weather data, while the areas highlighted in pink remains relatively invariant. At this stage, no relevant information on the measurement environments is provided, but from the plot, it can be reasonably inferred that the device has been relocated between indoor and outdoor environments during the 14-day period. This find is also studied later in the automatic anomaly detection in Section 3.3.

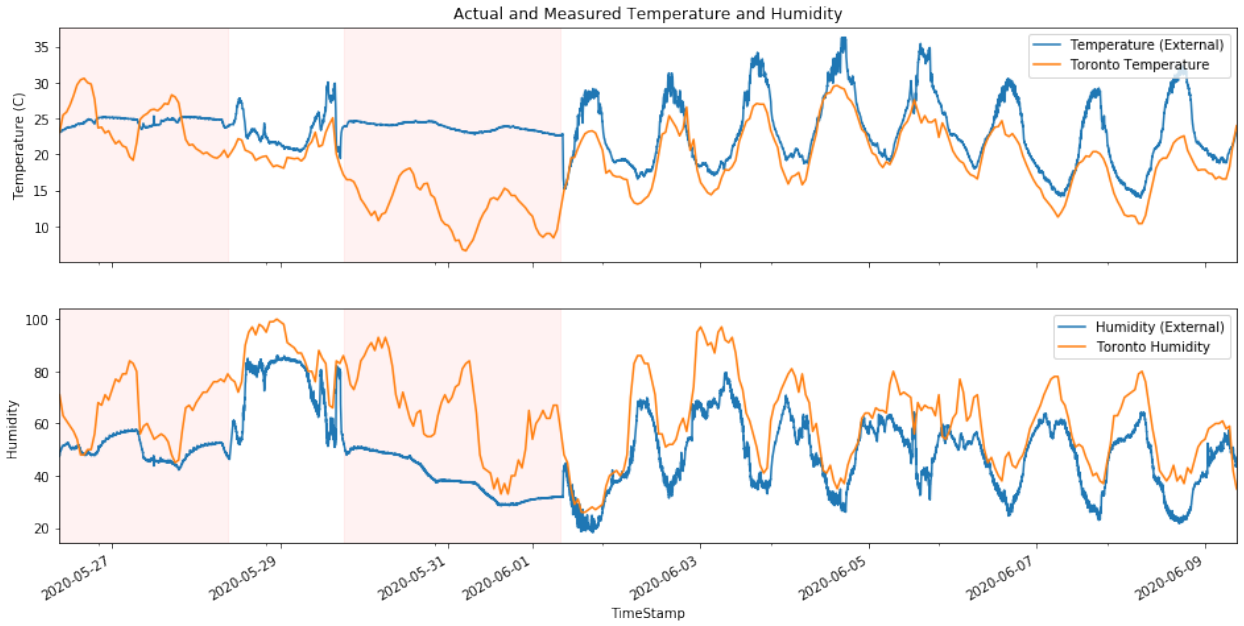

Figure 2: Temperature (External) and Humidity (External) overlaid with Toronto weather data. The pink highlighted areas remain relatively invariant compared to others.

2.2 Correlation Between Features

Correlation is a technique for investigating the relationship between two quantitative, continuous variables in order to represent their inter-dependencies. Among different correlation techniques, Pearson's correlation is the most common one, which measures the strength of association between the two variables. Its correlation coefficient scales from -1 to 1, where 1 represents the strongest positive correlation, -1 represents the strongest negative correlation and 0 represents no correlation. The correlation coefficients between each pair of the dataset are calculated and plotted as a heatmap, shown in Table 2. The scatter matrix of selected features is also plotted and attached in the Appendix section.

The first thing to be noticed is that PM 1, PM 2.5 and PM 10 are highly correlated with each other, which means they always fluctuate in the same fashion. Ozone is negatively correlated with Carbon Dioxide and positively correlates with Temperature (Internal) and Temperature (External). On the other hand, it is surprising not to find any significant correlation between Temperature (Internal) and Temperature (External), possibly due to the superior thermal insulation of the instrument. However, since there is no relevant knowledge provided, no conclusions can be made on the reasonability of this finding. Except for Ozone, Temperature (Internal) is also negatively correlated with Carbon Dioxide, Hydrogen Sulfide, and the three particulate matter measures. On the contrary, Temperature (External) positively correlates with Humidity (Internal) and three particulate matter measures, while negatively correlates with Humidity (External), just as what can be found from the time series plots in Figure 1.

Table 2: Heatmap of Pearson's Correlation Coefficients between Features.

3. Anomaly Detection

In this section, various anomaly detection methods are examined based on the dataset. The data come without labels, so there is no knowledge or classification rule is provided to distinguish between "system faults", "external events" and others. Any details of the instrument and experiments are not provided either. Therefore, the results in this section might deviate from expectations, but I am trying my best to make assumptions, define problems and then accomplish them based on my personal experience. The section consists of three parts: Point Anomaly Detection, Collective Anomaly Detection and Clustering.

3.1 Point Anomaly Detection (System Fault)

Point anomalies, or global outliers, are those data points that are entirely outside the scope of the usual signals without any support of close neighbors. They are usually caused by human or system error and they need to be removed during data cleaning for better performance in predictive modeling. In this dataset, by assuming the "system faults" are equivalent to such point anomalies, there are several features that are worth examining, such as the examples shown below in Figure 3.

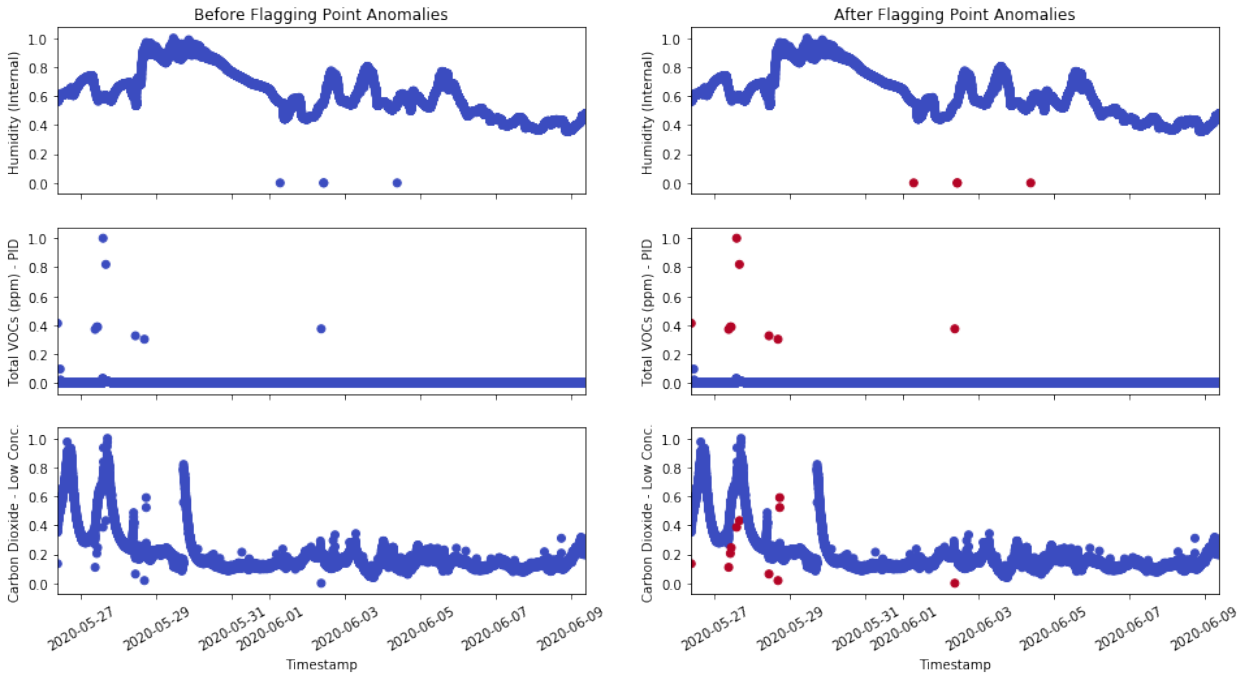

Figure 3: Time series signals before (left) and after (right) automatic flagging. The point anomalies (outliers) are marked as red. From top to bottom: Humidity (Internal), Total VOCs and Carbon Dioxide.

Here, from Humidity (Internal) to Total VOC to Carbon Dioxide, they each represents a distinct complexity of point anomaly detection tasks. In the first one (Figure 3, row 1), three outliers sit on the level of 0, so a simple boolean filter can do its job of flagging these data points. In the second one (Figure 3, row 2), the outliers deviate significantly from the signal that we are interested in, so linear thresholds can be used to separate out the outliers. Both cases are easy to implement, because they can be done by purely experience-based methods. When it comes to the third case (Figure 3, row 3), it is not possible to use a linear threshold to separate out the outliers since even if they deviate from its neighbors, the values may not be as great as the usual signals at other time points.

For such cases, there are many ways to approach. One of the simple ways is to calculate the rolling mean or median at each time point and test if the actual value is within the prediction interval that is calculated by adding and subtracting a certain fluctuation range from the central line. Since we are dealing with outliers in our case, the rolling median is more robust, so it is used in this dataset.

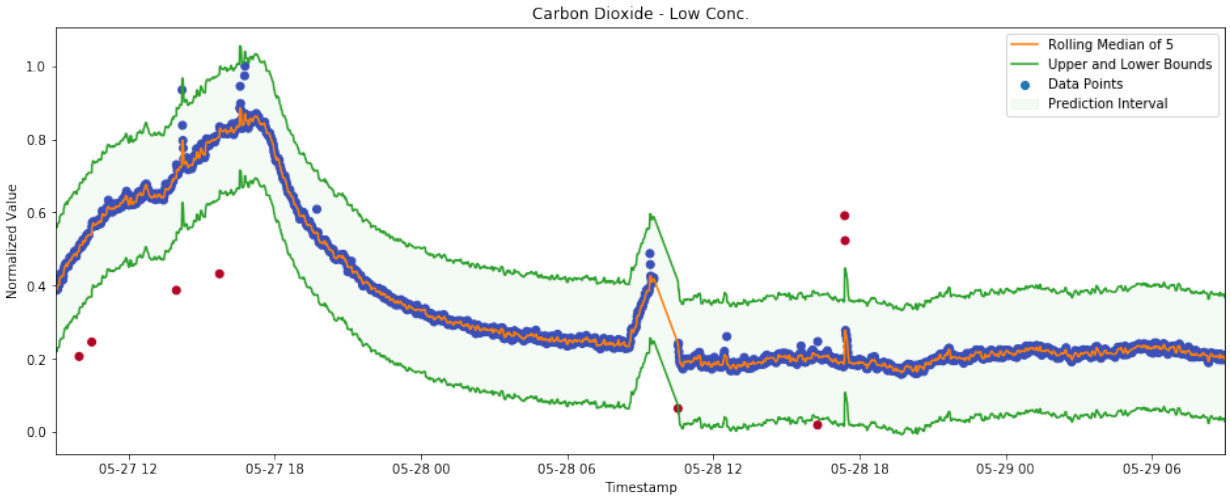

Figure 4: Zoomed-in time series plot of Carbon Dioxide, with its rolling median and prediction interval

From Figure 4, it is even clearer to demonstrate this approach: Equal-distance prediction intervals are set based on the rolling median. Here, the rolling window is set to 5, which means that for each data point, we take four of its closest neighbors and calculate the median as the center of prediction. Then a prediction interval of ± 0.17 is padded around the center. Any points outside are considered as outliers.

It is straightforward and efficient to detect the point anomalies using this approach. However, it has deficiencies and may not be reliable enough to deal with more complex data. In this model, there are two parameters: rolling window size and prediction interval size, both of which are defined manually through experimentations with the given data. We are essentially solving the problem encountered when going from case 2 to case 3 as seen in Figure 3 by enabling the self-adjusting capability to the classification boundary between signal and outliers according to time. However, the bandwidth is fixed, so it becomes less useful in cases when the definition of point anomalies changes with time. For example, the definition of a super cheap flight ticket might be totally different between a normal weekday and the holiday season.

That is when machine learning comes into play, where the model can learn from the data to adjust the parameters by itself when time changes. It will know when a data point can be classified as point anomaly or not at a specific time point. However, a supervised machine learning model cannot be constructed on this task for this dataset since the premise is to have labeled data, and this dataset does not. It would be still be suggested to go along this path in the future because with such model, the most accurate results will be generated no matter how complex the system or the data are.

Even though supervised learning methods would not work in this task, there are unsupervised learning techniques that can be useful such as clustering, also discussed in subsection 3.3. Clustering can group unlabeled data by using similarity measures, such as the distance between data points in the vector space. Then the point anomalies can be distinguished by selecting those far away from cluster centers. However, in order to use clustering for point anomaly detection in this dataset, we have to follow the assumption that external events are defined with the scope of multiple features instead of treating every single time series separately. More details on this are discussed in the following two subsections 3.2 and 3.3.

3.2 Collective Anomaly Detection (External Event)

If we define the "system fault" as point anomaly, then there are two directions to go with the "external event". One of them is to define it as a collective anomaly that appears in every single time series signals. The idea of collective anomaly is on the contrary to point anomaly. Point anomalies are discontinued values that are greatly deviated from usual signals while collective anomalies are usually continuous, but the values are out of expectations, such as significant increase or decrease at some time points. In this subsection, all 11 features are treated separately as a single time series. Then the task is to find the abrupt changes that happen in each of them.

For such problem, one typical approach is to tweak the usage of time series forecasting: we can fit a model to a certain time period before and predict the value after. The actual value is then compared to see if it falls into the prediction interval. It is very similar to the rolling median method used in the previous subsection, but only the previous time points are used here instead of using neighbors from both directions.

There are different options for the model as well. Traditional time series forecast models like SARIMA is a good candidate, but the model may not be complex enough to accommodate the "patterns" that mentioned in Section 2.1 and Section 3.3. Another option is to train a supervised regression model for the time series, which is quite widely used nowadays.

The idea is simple: Features are extracted from the time series using the concept of the sliding window, as seen in Table 3 and Figure 5. The sliding window size (blue) is set to be the same as the desired feature number k. Then for each data point (orange) in the time series, the features are data point values from its lag 1 to lag k before. As a result, a time series with N samples is able to be transformed into a table of N-k observations and k features. Next, by implementing the concept of "forward chaining", each point $\hat{y}_k$ is predicted by the regression model trained using observations from indices of 0 to k-1. In addition to the main regression model, two more quantile regressors are trained with different significance levels to predict the upper and lower bounds of the prediction interval, with which we are able to tell the actual value is above or below the interval band.

Table 3: Feature extraction algorithm through sliding window

Index	X_1 (Lag k)	…	X_{k-1} (Lag 2)	X_k (Lag 1)	y
0	$f(t_0)$	…	$f(t_{k-1})$	$f(t_k)$	$f(t_{k+1})$
1	$f(t_1)$	…	$f(t_k)$	$f(t_{k+1})$	$f(t_{k+2})$
2	$f(t_2)$	…	$f(t_{k+1})$	$f(t_{k+2})$	$f(t_{k+3})$
…	…	…	…	…	…
n	$f(t_n)$	…	$f(t_{k+n-2})$	$f(t_{k+n-1})$	$f(t_{k+n})$

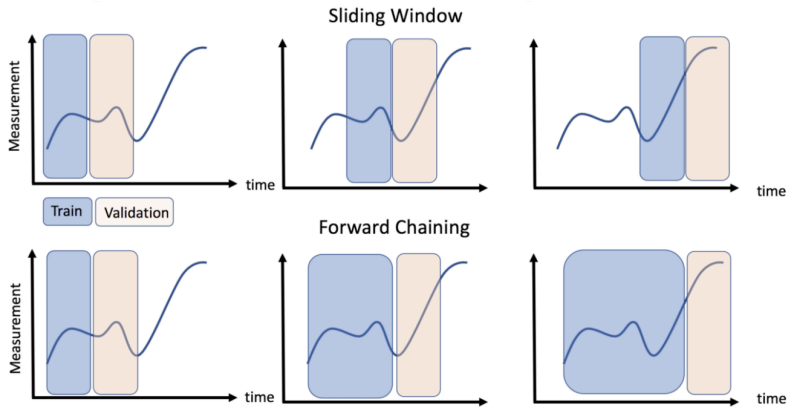

Figure 5: Concepts of sliding window and forward chaining [2]. The features and targets are extracted from the time series using sliding window. The training process is based on forward chaining.

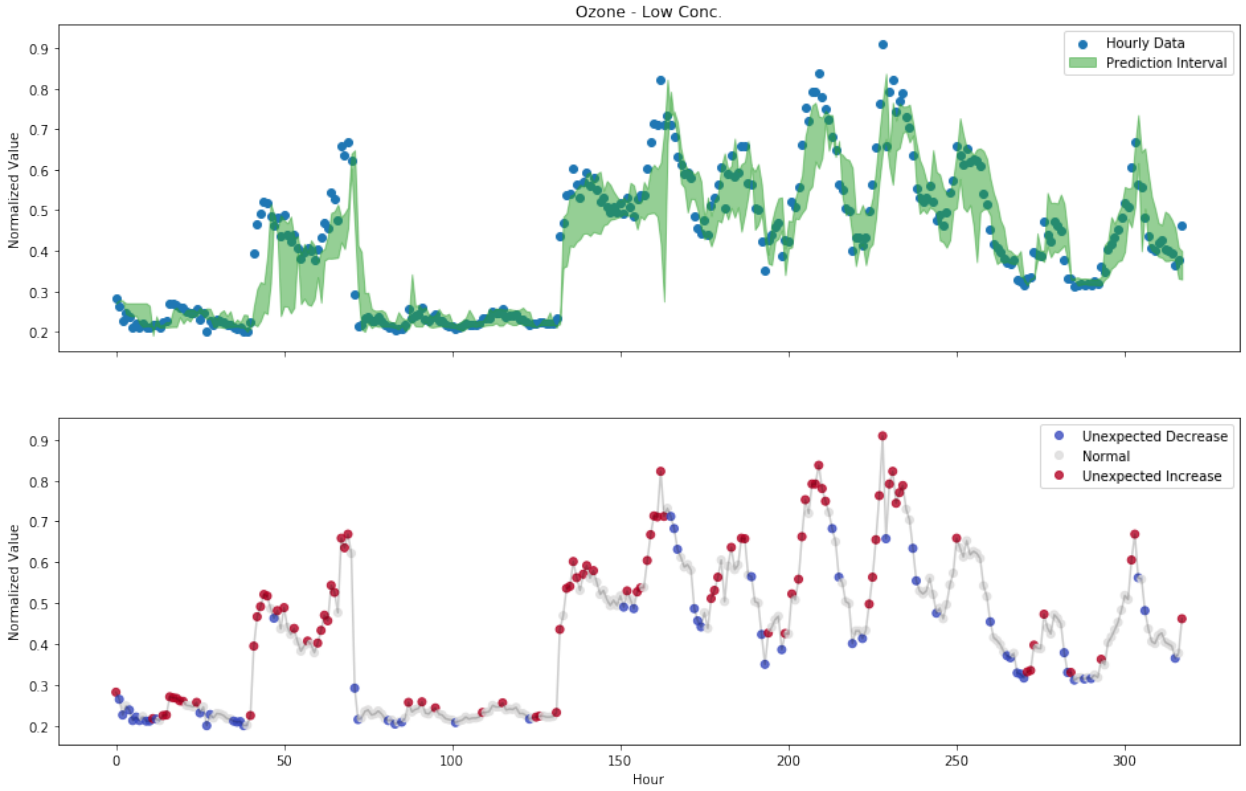

Figure 6: Flagging result on the Ozone time series data (hourly sampled) by applying the quantile regressor method. Top: data and prediction interval; Bottom: data with flags showing above or below the prediction interval.

This method is applied to the given dataset, and an example is shown above as a result in Figure 6. The Ozone time series is hourly sampled for faster training speed and the features are extracted and fed into three Gradient Boosting Regressor models (1 main and 2 quantile regressors) using Scikit-Learn. The significance levels are chosen so that the prediction level represents a 90% confidence interval (shown as green in Figure 6 Top). The actual values are then compared with the prediction interval and flagged with red (unexpected increase) and blue (unexpected decrease) in Figure 6 Bottom.

The results may not be super impressive yet, because more work still needs to be done around regression model selections and hyperparameter fine-tunings. However, it is already showing its capability of fagging these abrupt increases and decreases in all these spikes. One great thing about using machine learning models is that the model learns and evolves by itself when feeding with data. From Figure 6 Bottom, it can be seen that the last three hills (after about 240 hours) have fewer flagged points than the previous ones. It is not only because the magnitudes are smaller, but also due to the fact that the model is learning from the previous experience and start to adapt to the "idea" that it is now in "mountains" and periodic fluctuations should be expected. Therefore, it is not hard to conclude that the model performance can get better and better if more data instances are fed.

Beyond this quantile regression model, deep learning models such as LSTM might be able to achieve better performance. Long Short Term Memory (LSTM) is a specialized artificial Recurrent Neural Network (RNN) that is one of the state-of-the-art choices of sequence modeling due to its special design of feedback connections. However, it takes much longer time and effort to set up and fine-tune the network architecture, which exceeds the time allowance of this project, so it is not included as the presentable content in this report.

Again, on the other hand, as mentioned in previous sections, the provided data does not come with labels showing which data points are considered as anomalies. It does cause difficulties in the collective anomaly detection task discussed in this subsection by limiting the model choices and performance. In the future, if some labels are provided, it will become a semi-supervised or supervised learning problem, so that it will become easier to achieve better results.

3.3 Clustering (External Event)

As mentioned in subsection 3.2 above, recognizing the "external events" can be approached in two directions: one is to treat every time series separately and monitor any unexpected changes happened to the sensor signals, and the other one is to assume the events affect multiple features at the same time so that we hope to distinguish the events by looking at the unique characteristics shown in different features. In this case, if we have labeled data, it would be a common classification problem, but even without labels, we are still able to approach by clustering.

Clustering is an unsupervised machine learning technique that finds similarities between data according to the characteristics and groups similar data objects into clusters. It can be used as a stand-alone tool to get insights into data distribution and can also be used as a preprocessing step for other models. There are many distinct methods of clustering, and here two of the most common algorithms are used: K-Means and DBSCAN.

K-means is one of the partitioning methods used for clustering. It randomly partitions objects into nonempty subsets and constantly adding new objects and adjust the centroids until a local minimum is met when optimizing the sum of squared distance between each object and centroid. On the other hand, Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a density-based method where a cluster is defined as the maximal set of density-connected points [3].

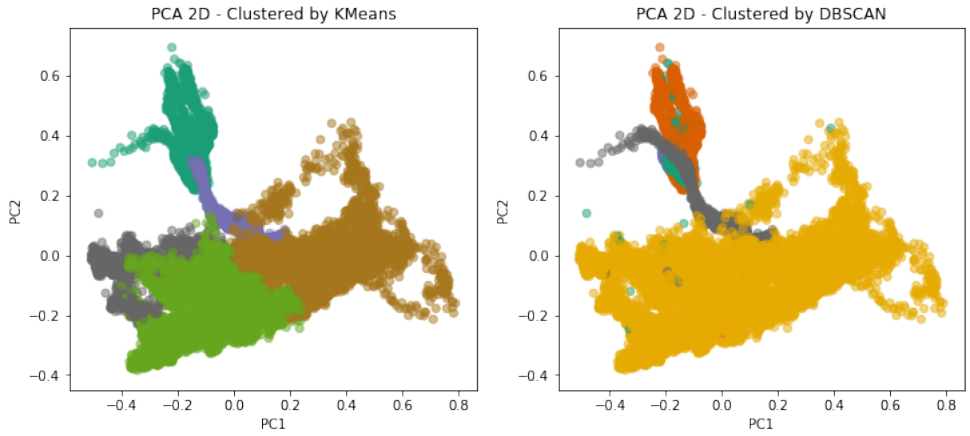

Figure 7: Clustering results projected to the first 2 principal components. Left: K-Means; Right: DBSCAN

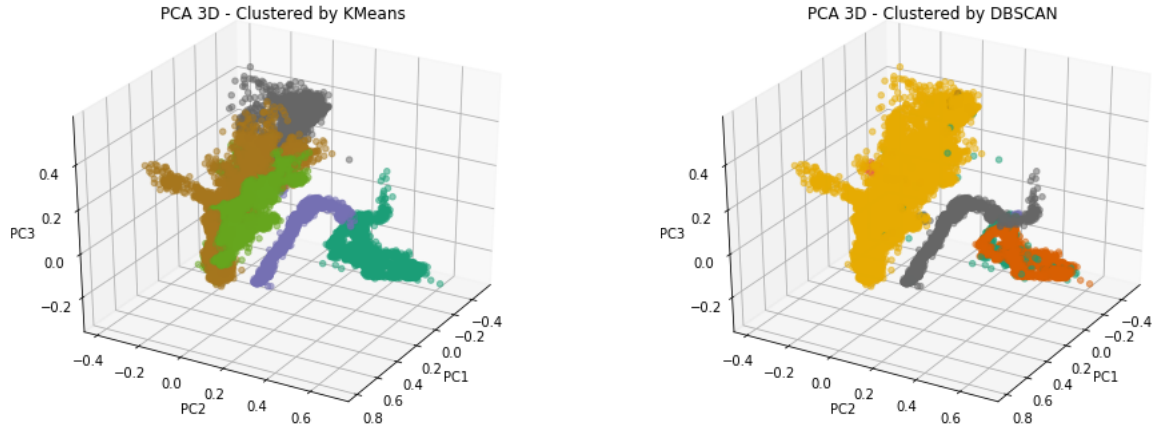

Figure 8: Clustering results projected to the first 3 principal components. Left: K-Means; Right: DBSCAN

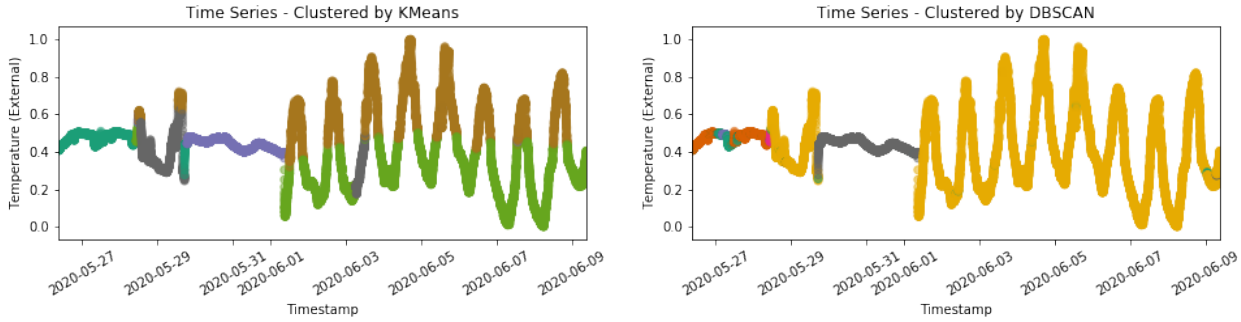

Figure 9: Clustering results on time series of Temperature (External). Left: K-Means; Right: DBSCAN

Principal Components Analysis (PCA) is a dimension reduction technique that creates new uncorrelated variables in order to increase interpretability and minimize information loss. In this project, after applying K-means and DBSCAN algorithms on the normalized data, PCA is performed and the clustering results are plotted in both 2D (Figure 7) and 3D (Figure 8) using the first 2 and 3 principal components. In addition, to view the clustering results from another perspective, the labeled time series plots are made, and the Temperature (External) plot is shown in Figure 9 as an example.

From the plots, it can be clearly seen that both methods are able to distinguish between the indoor/outdoor pattern changes that mentioned in section 2.1. The main difference is that the partition-based K-means method is more sensitive to the magnitude changes caused by day/night alternation. Many variables in the dataset are subject to such obvious sinusoidal changes that happen at the same time, including Temperature (External and Internal), Humidity (External and internal) and Ozone. K-means tends to treat peaks and valleys differently. On the other hand, density-based DBSCAN cares less about the magnitude difference, but pays more attention to the density distributions. Therefore, it clusters the whole sinusoidal part as one mass cloud as seen in Figure 7 and Figure 8.

It is not possible to comment on which clustering method is better than the other at this stage, because they are distinctive enough to function for different interests. If we are more interested in treating the high-low portions of the sinusoidal signals differently, K-means can be used; if we only want to distinguish between indoor/outdoor mode, then DBSCAN is better. In addition, since it is an unsupervised learning task, there is no way of quantifying the performance between models except for visualizing and judging by experience. In the future, if some labeled data is provided, the results can be turned into a semi-supervised learning task, and more intuitions can be gained toward model selection.

4. Conclusion

In this report, I briefly walk through the approaches and findings of exploratory data analysis and correlation analysis, as well as the constructions of three distinct modeling pipelines that used for point anomaly detection, collective anomaly detection, and clustering.

In the exploratory data analysis section, the sensor reading time intervals are found to vary severely. Even if most of them are contained around a minute, the inconsistency problem is still worth looking into due to the fact that it might lower the efficiency and performance of analytical tasks. The measuring environment is found subject to changes from the time series plots and later reassured by aligning with the actual Toronto weather data as well as the clustering results. In addition, the correlations between features are studied and exemplified. A few questions are raised such as the strange relationship between Temperature (Internal) and Temperature (External), which needs to be studied through experiments or the device itself.

In the anomaly detection section, since “system fault” and “external event” are not clearly defined, the project is split into three different tasks. Point anomalies are defined as severely deviated and discontinued data points. The rolling median method is used here to successfully automate the process of labeling such point anomalies. Collective anomalies, on the other hand, are defined as the deviated collection of data points, usually seen as abrupt increases or decreases. This task is accomplished by extracting features from time series data and then training of regression models. Clustering is also performed on the dataset using K-mean and DBSCAN, both of which play to their strength and successfully clustered data by leveraging their similar and dissimilar characteristics.

All of the anomaly detection models introduced in this project are only prototypes without extensive model sections and fine-tunings. There are great potentials for each of them to evolve into better forms if putting more effort and through gaining more knowledge of the data. For point anomalies, there are many more machine-learning-based outlier detection techniques such as isolation forest and local outlier factors to accommodate for more complex data forms. For collective anomaly,

state-of-the-art LSTM is worth putting effort into, especially in time series data and sequence modeling. For clustering, there are many other families of methods, such as hierarchical and grid-based clustering. They are capable of achieving similar great performance.

At the current stage, these future directions are advised based on the premise of no labeled data. If experienced engineers or scientists are able to give their insights, and efforts can be put into labeling the dataset, more exciting progress will surely be made by transforming the tasks into semi-supervised or supervised learning problems, where more tools will be available to choose.

References

[1] Canda Weather Stats: https://www.weatherstats.ca/

[2] Time Series Machine Learning Framework: https://towardsdatascience.com/time-series-machine-learning-regression-framework-9ea33929009a

[3] J. Han, M. Kamber and J. Pei, *Data Mining: Concepts and Techniques*, 3rd edition, 2011.

Appendix – Scatter Matrix of Selected Features