Clustering Sensor Data from Building Automation Systems to Assist Fault Detection

Shanghao Chen

Advisors: Steven Bergner, Ken Lockhart

August 29, 2018

Contents

1 Introduction

This report includes four aspects below:

1. Introducing the concept of time-series data clustering and the relevant algorithms used in the current project (2).
2. Comparison of different clustering methods when applied to all temperature data (3).
3. Visual evaluation of room temperature (RT) and supply air temperature (SAT) involving an interactive visualization to explore clustering results (4).
4. Potential future work (5).

2 Background on Time-series Data Clustering

Clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters). Clustering of time-series data is mostly utilized for the discovery of interesting patterns in time-series datasets, which can help to discover unusual and unexpected patterns. Time-series data clustering usually has four components: dimensionality reduction methods (2.1), distance measures (2.2), clustering algorithms (2.3) and clustering evaluation measures (2.4). In this section, the relevant technologies used in the project are discussed and compared.

2.1 Dimensionality Reduction Methods

A time-series that is observed at discrete time steps within a given interval can be represented as a vector, where each dimension corresponds to an observation at a particular time. Dimensionality reduction represents the raw time-series in another space by transforming the time-series to a lower dimensional space, where in some cases a dimension can be interpreted to indicate the presence of a particular feature. Such a reduction lowers memory costs and may improve speed and robustness of subsequent clustering. In fact, it is a trade-off between speed and quality and some clustering parameters may be tuned to obtain a proper balance point between quality and execution time. In this project, tow techniques, PCA (2.1.1) and T-SNE (2.1.2), have been used.

2.1.1 Principal Component Analysis

Principal Component Analysis (PCA) is a linear method that can be used to project a large set of variables to a smaller set that still contains most of the information in the large set, as measured in terms of preserved variance.

Details of the algorithm can be viewed on Wikipedia.
https://en.wikipedia.org/wiki/Principal_component_analysis

2.1.2 T-Distributed Stochastic Neighbor Embedding

T-distributed stochastic neighbor embedding (t-SNE) is a non-linear technique for dimensionality reduction that is particularly well suited for the visualization of high-dimensional datasets.

Details of the algorithm can be viewed on Wikipedia.
https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding

2.2 Distance Measures

The definition of a distance measure between time-series is critical since clustering algorithms are based on a notion of distance among vectors or points (each vector representing a time-series). In this subsection, Euclidean Distance (2.2.1), Normalized Euclidean Distance (2.2.2), Euclidean distance (DTW) (2.2.3) and Normalized Euclidean Distance (DTW) (2.2.4) are compared to each other's advantages and disadvantages (2.2.5).

2.2.1 Euclidean Distance

The Euclidean distance is the "ordinary" straight-line distance between two points in Euclidean space. The calculation formula is as follows:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

A key problem for Euclidean distance is its sensitivity to "alignment of values", i.e. that corresponding values p and q are found at the same index i, which, for instance, is not the case if the two series are shifted by some delay with respect to each other.

2.2.2 Normalized Euclidean Distance

The normalized Euclidean distance rescales mean and variance of data values in advance and then calculates the Euclidean distance between two points.

$$d(p',q') = \sqrt{\sum_{i=1}^{n}(q_i' - p_i')^2}$$

$$mean(p') = mean(q') = 0, sd(p') = sd(q') = 1$$

2.2.3 Euclidean Distance through Dynamic Time Warping

Dynamic time warping (DTW) is one of the algorithms for measuring the distance between two temporal sequences, which may vary in speed. Euclidean Distance through Dynamic Time Warping is using dynamic programming to find an optimal Euclidean Distance match between two given sequences.

The time complexity of DTW algorithm is $O(NM)$, where N and M are the lengths of the two input sequences.
The following figure describes the difference between Euclidean Distance and Euclidean Distance through DTW.

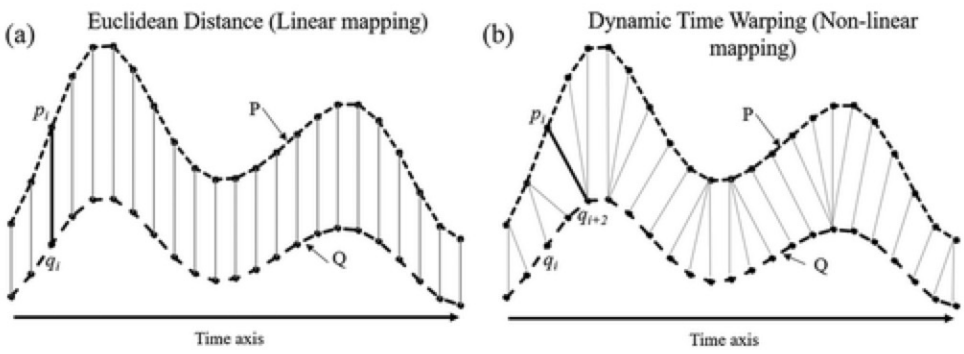

Figure 1: Difference Between Euclidean Distance and Dynamic Time Warping

2.2.4 Normalized Euclidean Distance through Dynamic Time Warping

This distance measure is using dynamic programming to find an optimal Normalized Euclidean Distance match between two given sequences.

2.2.5 Comparison with different Distance Measures

The following series curves are used to compare different distance formulas. There are 100 data points evenly distributed in the range of 0 to 50. There is a isolated singularity in $ts4$, i.e. $ts4[x = 10] = 50$.

$$ts1 = 3 * sin(x/3) + 3.5 \qquad ts2 = 3 * sin(x/3) + 2.5$$

$$ts3 = 3 * sin((x - 4)/3) + 3.5 \qquad ts4 = 3 * sin(x/3) + 3.5$$

The four series curves are shown in the following figure.

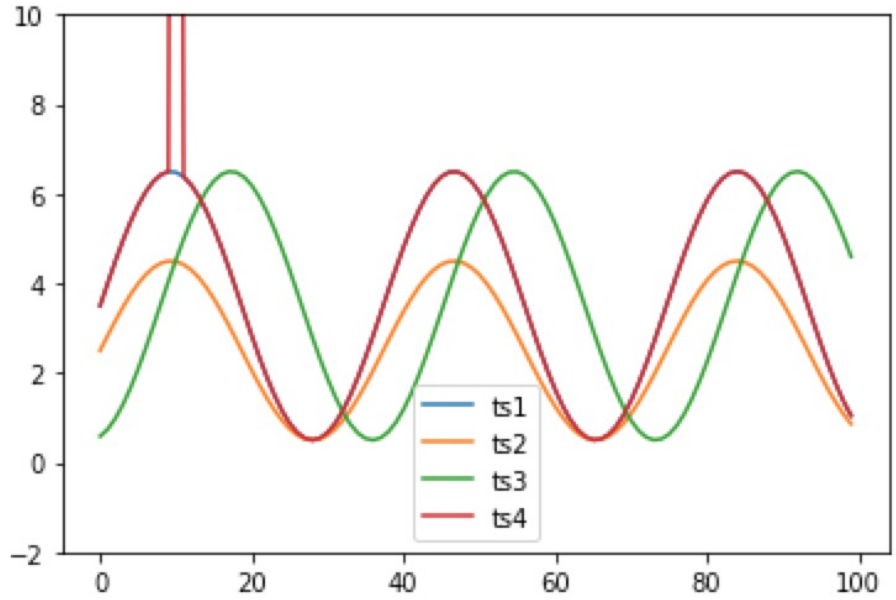

Figure 2: Dynamic Time Warping Example

The following table is obtained by using $ts1$ as a datum curve, comparing the other curves and its distance.

Approaches / Curves	Euclidean	N-Euclidean	Euclidean (DTW)	N-Euclidean (DTW)
ts2	18.27	17.81	11.27	12.54
ts3	37.98	29.51	7.95	14.14
ts4	50.00	2.22	7.07	1.48

Table 1: Comparison of Different Distance Measures

From the above table, we observe the following.

1. Euclidean distance is not suitable for calculating the distance of two temporal sequences that are not aligned.
2. In comparison with dynamic time warping, Euclidean Distance and Normalized Euclidean Distance is a somewhat sensitive to shifting.

3. Both Euclidean Distance and Normalized Euclidean Distance through DTW are good for measuring distance between two temporal sequences. The remaining problem for these measures is the computational cost of DTW.

2.3 Clustering Algorithms

In this subsection, different clustering algorithms, including DBSCAN (2.3.1), K-Means (2.3.2), K-Medoids (2.3.3) are presented.

2.3.1 DBSCAN

DBSCAN means Density-based spatial clustering of applications with noise, which finds core samples of high density and expands clusters from them. There is no need to set the number of cluster in advance and the most important parameters in DBSCAN are *eps* and *min_samples*.

The detail of the DBSCAN can be viewed in Wikipedia.
https://en.wikipedia.org/wiki/DBSCAN

2.3.2 K-Means

The algorithm works iteratively to assign each data point to one of the k groups based on the features that are provided. Data points are clustered based on feature similarity. The key parameter in k-means is k, the number of clusters that has to be chosen before clustering.

The detail of the k-means can be viewed in Wikipedia.
https://en.wikipedia.org/wiki/K-means_clustering

2.3.3 K-Medoids

The k-medoids algorithm is a clustering algorithm related to the k-means algorithm and the medoidshift algorithm.
The detail of the k-medoids can be viewed in Wikipedia.
https://en.wikipedia.org/wiki/K-medoids

2.4 Clustering Evaluation Measures

In this section, the evaluation methods for clustering algorithms used in our project are discussed. On possible use for such numerical evaluation measures, is to optimize them as an objective when choosing clustering parameters, such as k in k-means.

2.4.1 Silhouette Coefficient

The Silhouette Coefficient is a measure of how similar an object or vector is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from -1 to $+1$, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative values, then the clustering configuration may have too many or too few clusters.

The details of Silhouette Coefficient can be viewed in Wikipedia.
https://en.wikipedia.org/wiki/Silhouette_(clustering)

2.4.2 Sum of Squares Error

Sum of Squares Error (SSE) is the sum of the squares of residuals (deviations predicted from actual empirical values of data). It is a measure of the discrepancy between the data and an estimation model. A small SSE indicates a tight fit of the model to the data. It is used as an optimality criterion in parameter selection and model selection.
The details of Sum of Squares Error can be viewed in Wikipedia.
https://en.wikipedia.org/wiki/Residual_sum_of_squares

3 Clustering Result for Temperature Trend logs

In this section, different cluster approaches discussed in the previous section are applied to all temperature trend logs with sampling interval time of 300 seconds. The time range of the testing data is from March 1, 2018, to March 31, 2018, and the number of data samples for each trend log is 86400. The detailed conclusions of the T-SNE (3.1), PCA (3.2), DBSCAN (3.2.1), K-Means (3.2.2) and K-Medoids (2.3.3) can be found in the following subsections. At the end of this section, the results and conclusion of clustering all temperature logs with different sampling interval time are presented.

3.1 Visualize Data in Two Dimensions Through T-SNE

T-SNE (2.1.2) is particularly well suited for the visualization of high-dimensional datasets. The example in the figure below shows that it is hard to clearly separate different clusters in the 2-dimensional T-SNE graph since the graph provides little insight into the class structure of the data. Therefore, it seems infeasible to reduce 8640 dimensions of time-series data in 2 or 3 dimensions.

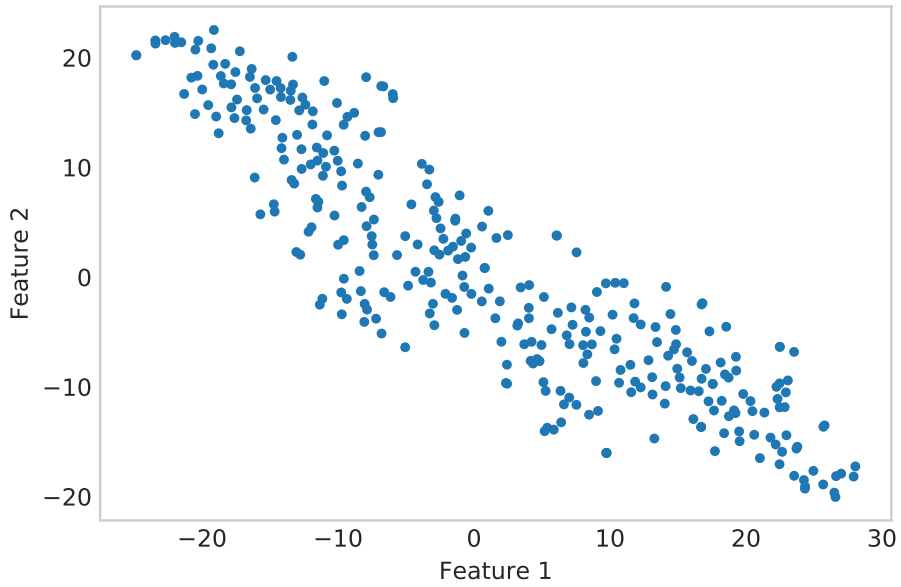

Figure 3: T-SNE Graph for One Month Temperature Data

3.2 Dimensionality Reduction By PCA

The graph below shows that PCA (2.1.1) can account for more than 80% variability in one-month of temperature data when the number of the components (reduced dimensions) is larger than 15. Since the initial dimensionality of each trend log is 8640, PCA can use only 0.1% dimensions to preserve more than 80% of variability from the original data. Consequently, it is a feasible approach to reduce dimensionality by PCA before clustering.

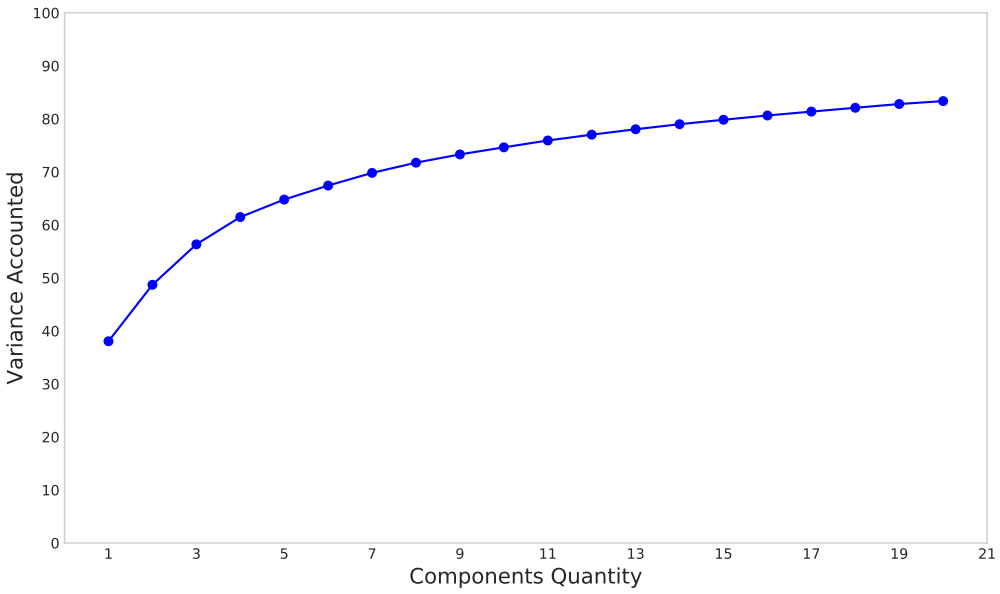

Figure 4: Dimensionality Reduction For One Month Data by PCA

3.2.1 DBSCAN (PCA, Normalized Euclidean Distance)

Experimental setup: Use Normalized Euclidean Distance as the distance measure, reduce dimensionality via PCA and apply DBSCAN (2.3.1) to cluster the data. The main process of the algorithm is as follows:

1. Extract the one month (from 2018-03-01 to 2018-03-31) temperature data with 300 seconds interval and 'Polled' units from MongoDB and transfer them to pandas Dataframe.
2. Clean the trend log with missing data larger than 30% and linearly interpolate the missing data for the remaining trend log.
3. Get the stable time series data ($\sigma <= 0.05$) as first stable cluster (28 samples).
4. Reduce the tend log data dimension via PCA
5. Compute pairwise correlation of columns to build the correlation matrix.
6. Use DBSCAN to cluster the data and adjust the parameters (min_samples, eps) to get the best result

The graph below shows the 28 items of stable (constant) curves, which are combined in cluster -1. Likely, all of these curves are set point trend logs.

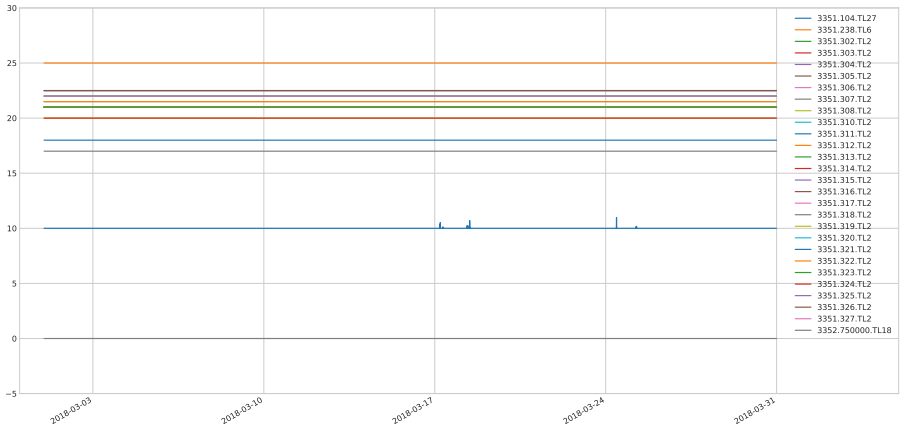

Figure 5: Cluster -1 (Set Point Trend Logs)

However, after testing *eps* from 0.05 to 0.8 with step 0.05 and *min_samples* from 2 to 7, it seems that DBSCAN still cannot cluster the rest of time-series data well, even when setting the number of components larger than 20, which means PCA has already retained more than 80% of variation. The test results are shown in the following table:

min_samples	eps_test	clusters	silhouette_score	Cluster Count
1	0.30	5	0.184224	{0: 319, 1: 3, 2: 1, 3: 1, 4: 1}
4	0.20	4	0.139233	{0: 238, -1: 75, 1: 4, 2: 4, 3: 4}
2	0.25	4	0.133219	{0: 305, 1: 3, -1: 11, 2: 2, 3: 4}
2	0.20	16	0.065817	{0: 246, 1: 2, 2: 3, -1: 37, 3: 4, 4: 2, 5: 2,...
3	0.20	9	0.065695	{0: 246, -1: 51, 1: 3, 2: 4, 3: 4, 4: 3, 5: 3,...
4	0.15	4	0.028022	{0: 135, -1: 166, 1: 16, 2: 4, 3: 4}
1	0.25	15	0.018565	{0: 305, 1: 3, 2: 1, 3: 1, 4: 1, 5: 1, 6: 1, 7...
3	0.15	13	-0.023856	{0: 142, -1: 132, 1: 3, 2: 17, 3: 3, 4: 3, 5: ...
2	0.05	6	-0.107476	{-1: 311, 0: 2, 1: 4, 2: 2, 3: 2, 4: 2, 5: 2}
3	0.10	14	-0.133961	{0: 5, -1: 256, 1: 4, 3: 6, 2: 3, 4: 8, 5: 3, ...
4	0.10	7	-0.152640	{0: 5, -1: 280, 4: 6, 1: 8, 2: 8, 3: 6, 5: 8, ...
5	0.10	6	-0.168446	{0: 5, -1: 285, 4: 6, 1: 7, 2: 8, 3: 6, 5: 8}

Figure 6: DBSCAN Test Results

From the above table, most of the data are in the same cluster when selecting the cluster number larger than 4. It indicates that DBSCAN is not the ideal cluster algorithms for long time-series data.

3.2.2 K-Means (Euclidean Distance and Normalized Euclidean Distance)

The process before using K-Means (2.3.2) cluster algorithm is essentially the same as for DBSCAN. The only difference is that there is no dimensionality reduction before applying the cluster algorithm. Then, testing K from 2 to 30 and set $internumber$ as 1500 and n_init as 50 based on Euclidean Distance and Normalized Euclidean Distance.
From the following the figures, the variation trend of the Sum of Squares Errors and Silhouette Coefficients under different K-values can be seen. It seems quite similar when testing different distance measures.

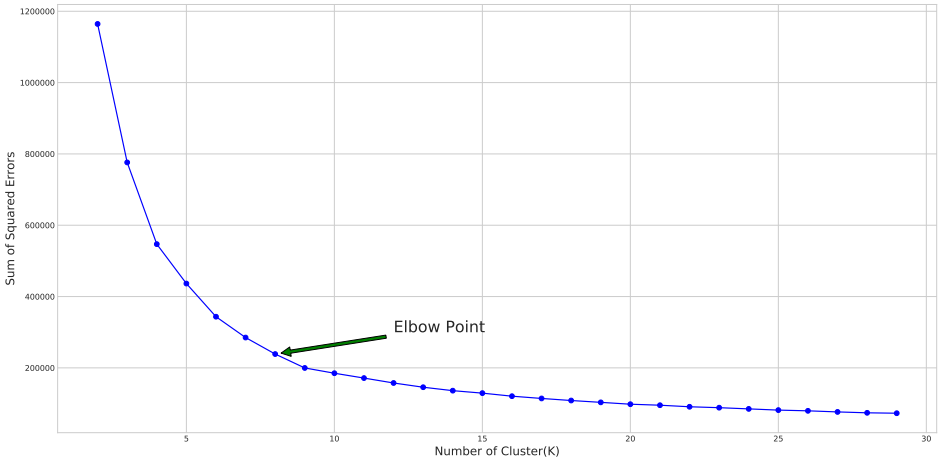

Figure 7: Sum of Squares Errors Test Result (Normalized Euclidean Distance)

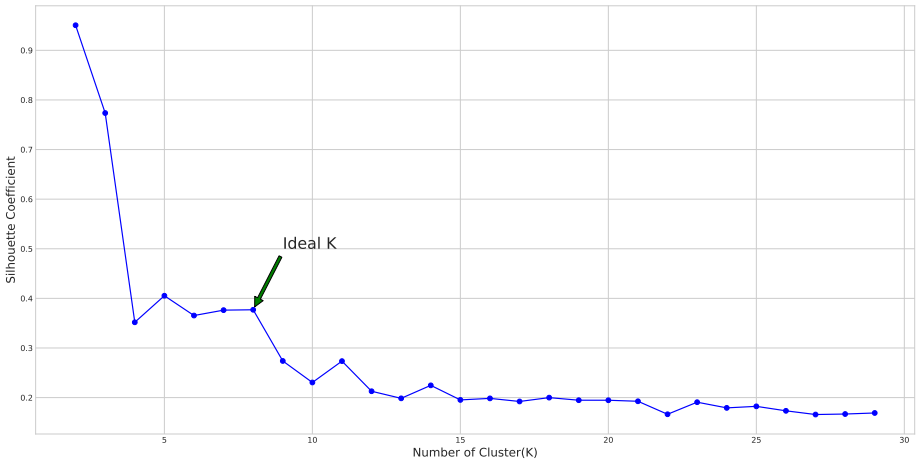

Figure 8: Silhouette Coefficient Test Result (Normalized Euclidean Distance)

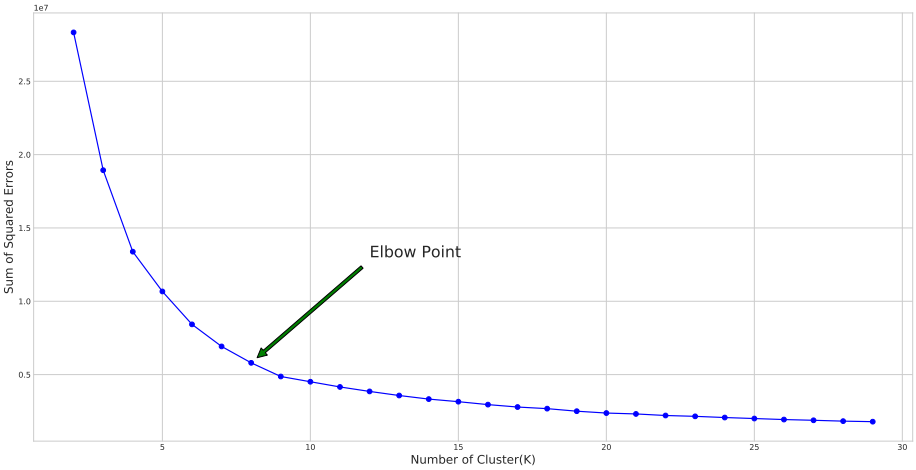

Figure 9: Silhouette Coefficient Test Result (Normalized Euclidean Distance)

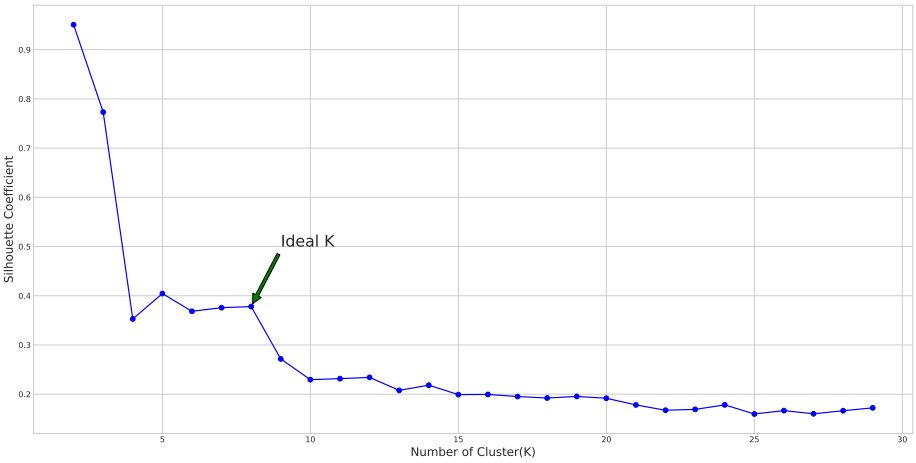

Figure 10: Silhouette Coefficient Test Result (Euclidean Distance)

From the above four figures, it seems that when K is larger than 8, SSE tends to converge and the silhouette coefficient has a relatively high value. Therefore, K equals 8 is an ideal number for clusters in K-means. Since the result is almost the same when using Euclidean Distance or Normalized Euclidean Distance, the following figures only display the detailed clusters when using Normalized Euclidean Distance results.

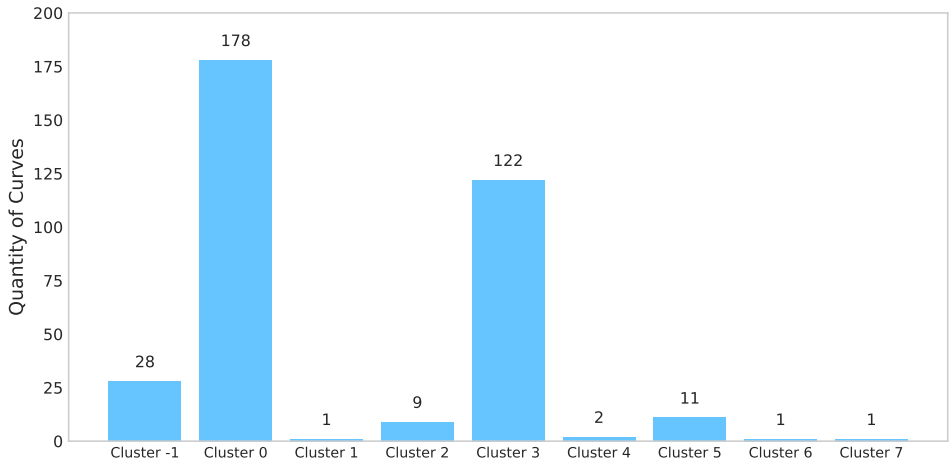

Figure 11: K-Means Clusters Distribution)

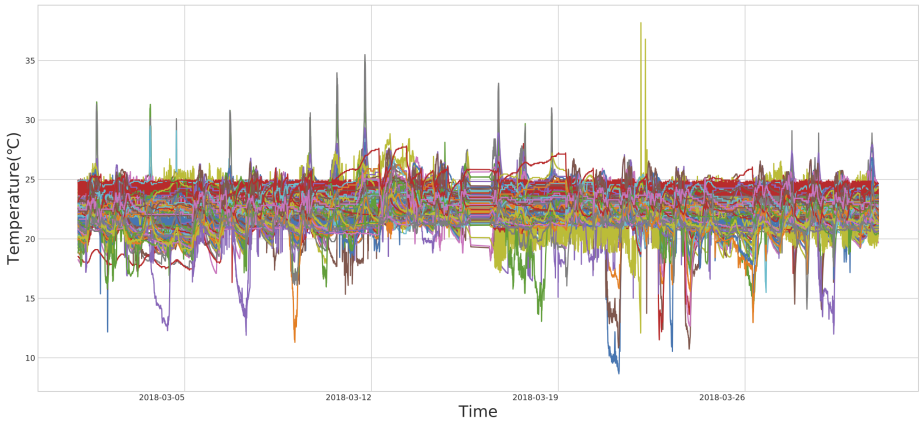

Figure 12: K-Means Clusters 0

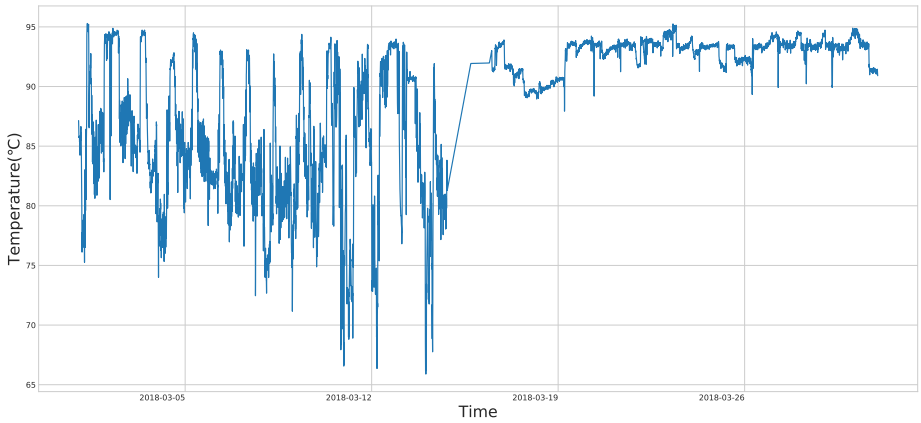

Figure 13: K-Means Clusters 1

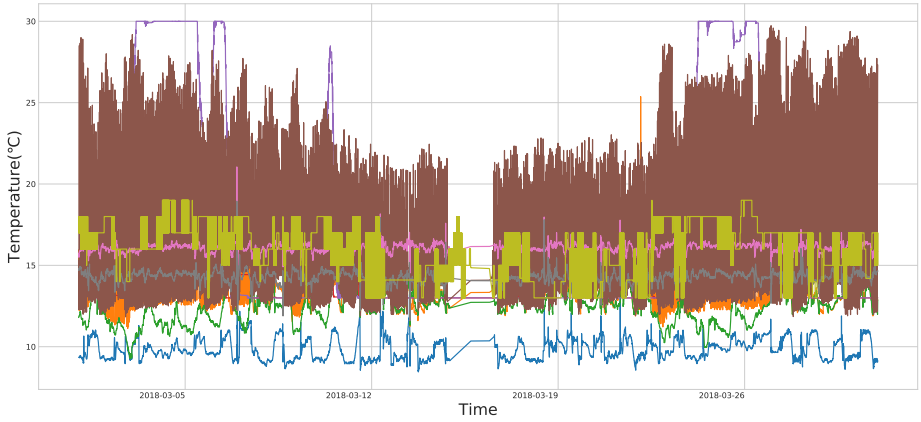

Figure 14: K-Means Clusters 2

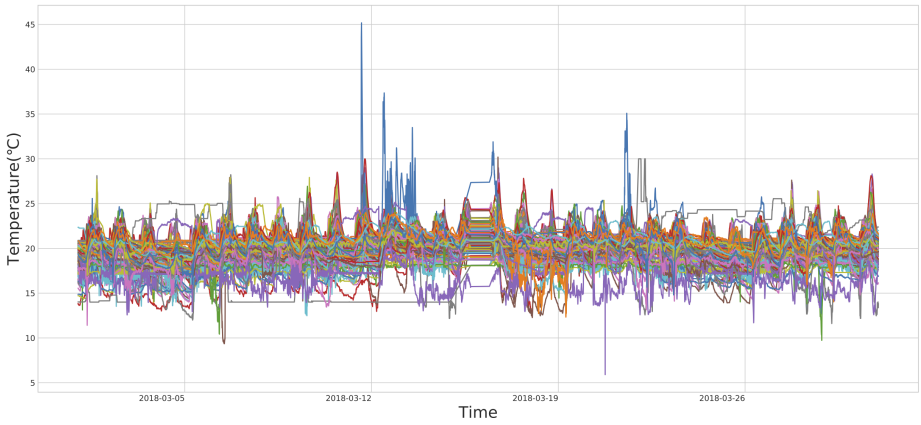

Figure 15: K-Means Clusters 3

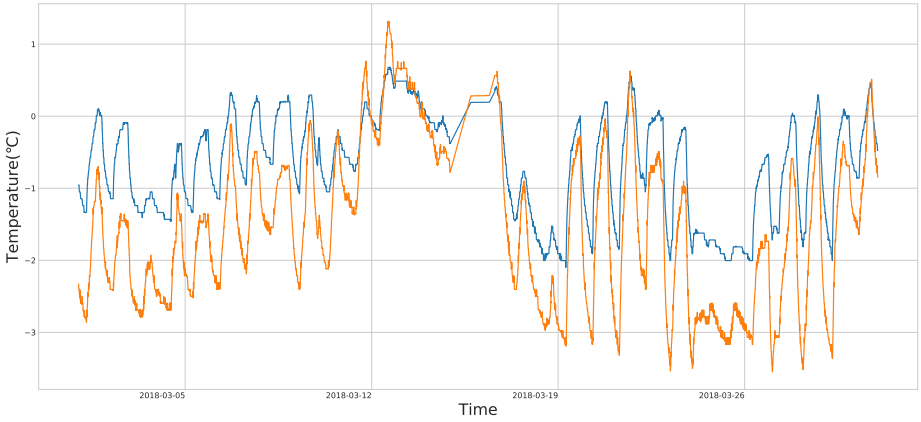

Figure 16: K-Means Clusters 4

Figure 17: K-Means Clusters 5

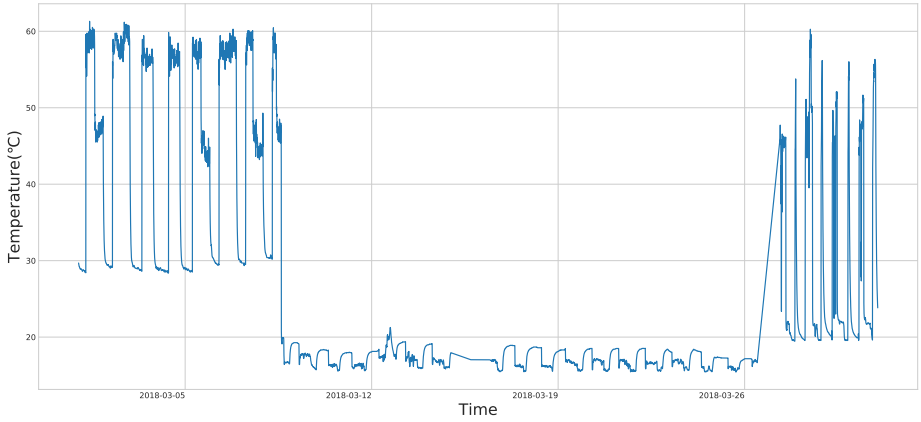

Figure 18: K-Means Clusters 6

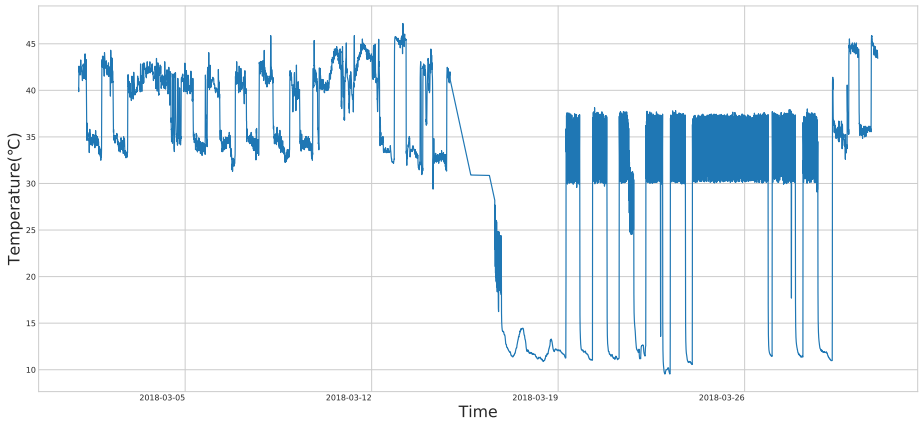

Figure 19: K-Means Clusters 7

Unfortunately, the distribution of clusters varies widely since most of the trend logs belong to cluster 0 and 3. To be specific, the clusters are still evenly distributed even when testing for K from 3 to 20. To investigate the reason, the following pie charts show the device distribution in different clusters (0, 2, 3, 5).

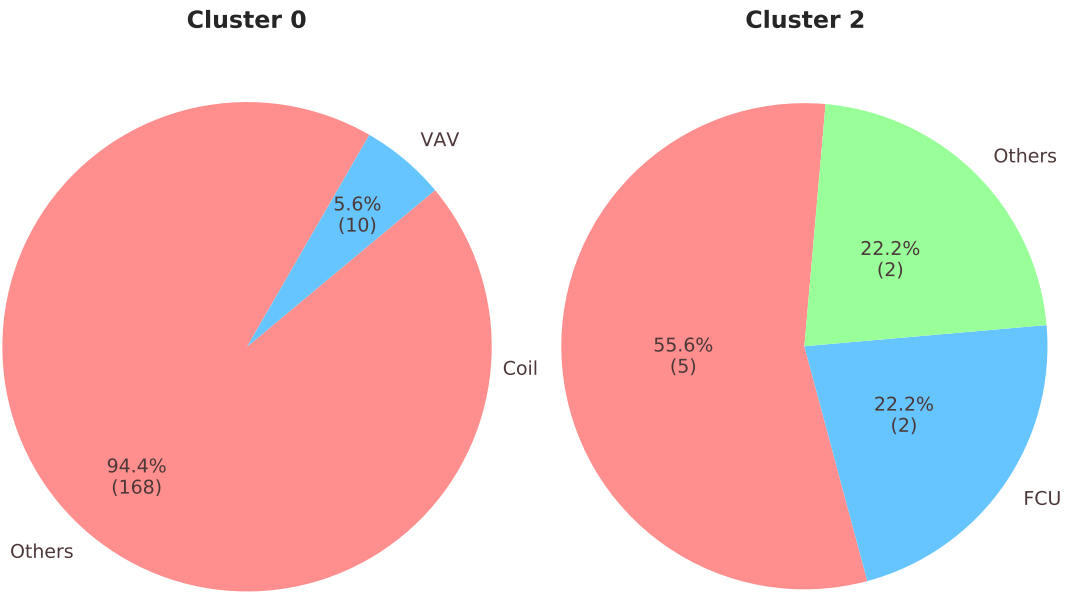

Figure 20: Devices Distribution 1

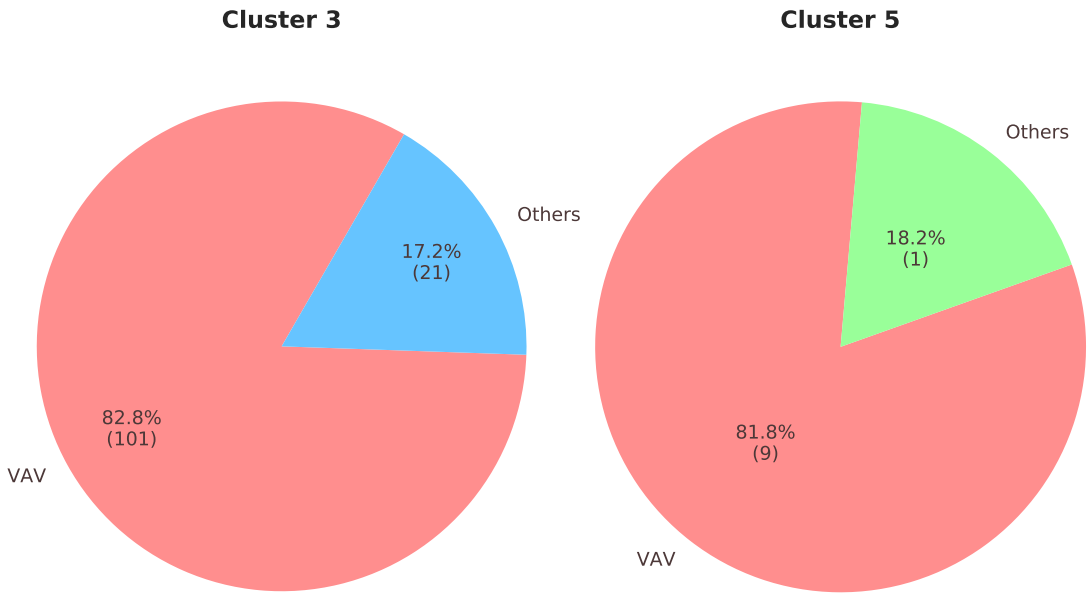

Figure 21: Devices Distribution 2

Most of the trend logs in cluster 0, cluster 3 and cluster 5 belong to Variable Air Volume (VAV). It seems that clustering all the temperature data directly simply groups the same types of devices in the same clusters when using K-Means. While this may be useful to confirm device types given in the meta data, clusters related to different trend log behaviour would more useful and could be obtained by first grouping the device types in before clustering each type separately.

3.3 K-Medoids (Normalized Euclidean Distance Through DTW)

It is not practical to use DTW in K-means because of the computation complexity. For instance, it takes more than 40 hours when calculating DTW for one-month of trend logs once. Therefore, only the clustering algorithm based on the distance matrix can be used. From the figures below, the clustering result from K-Medoids is better than K-Means from the last subsection. However, the results of K-Medoids always outputs inconsistent results for each test run, which is unreliable. However, the clusters distribution can be quite stable when testing K-Medoids several times. The following figures show good results from K-Medoids and $K = 14$.

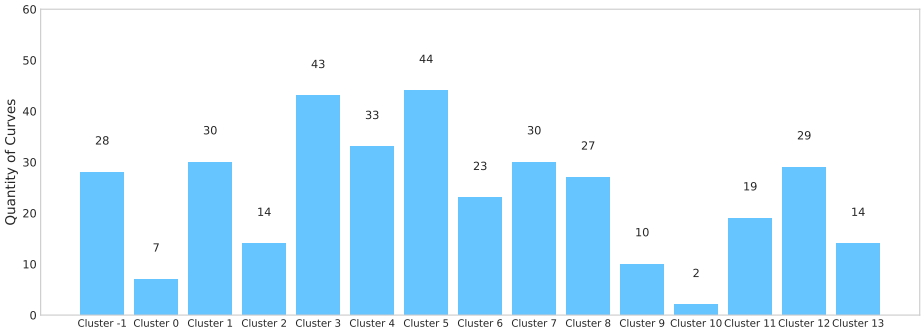

Figure 22: K-Medoids Clusters Distribution

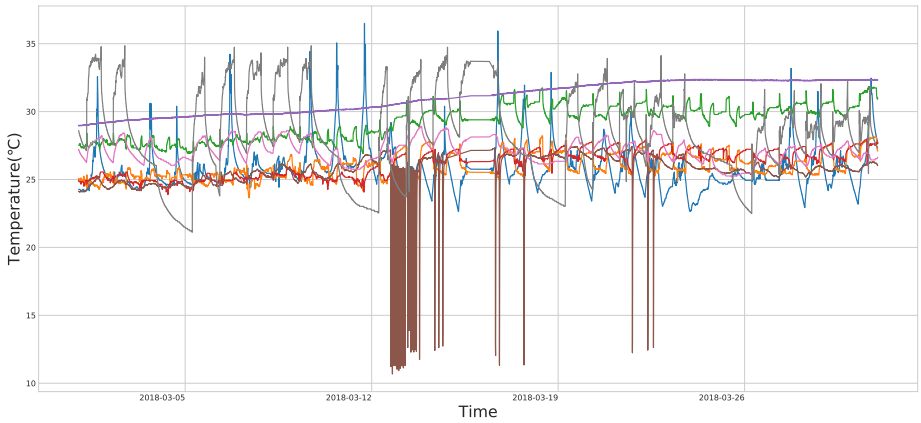

Figure 23: K-Medoids Cluster 0

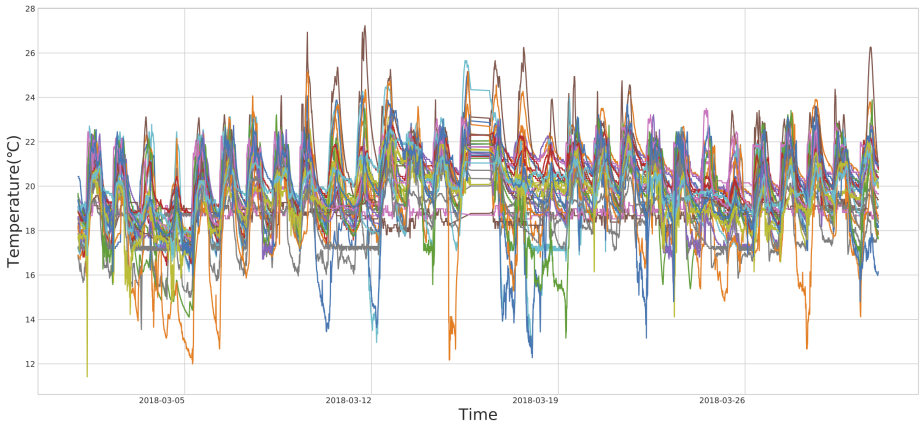

Figure 24: K-Medoids Cluster 1

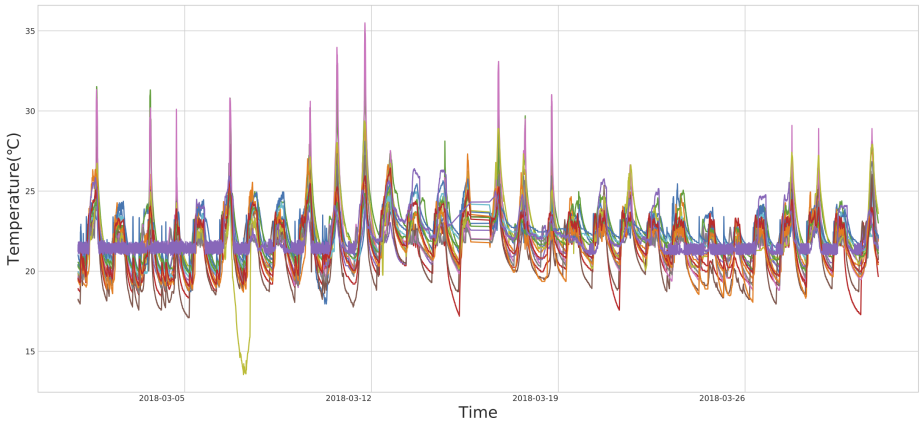

Figure 25: K-Medoids Cluster 2

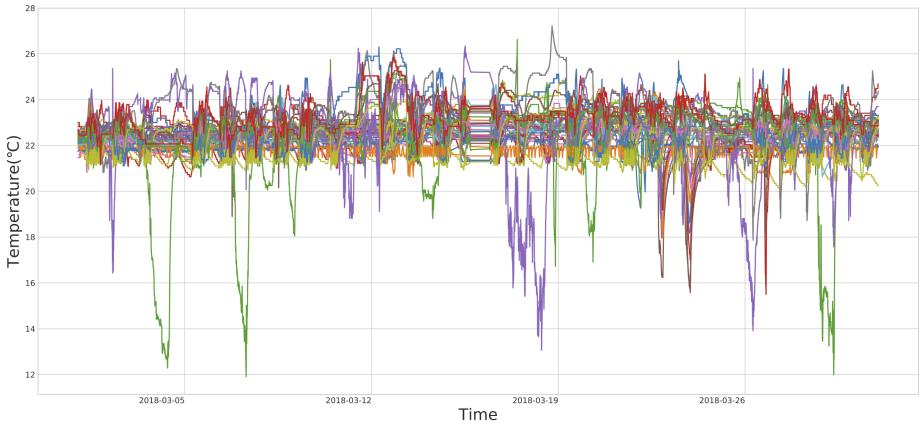

Figure 26: K-Medoids Cluster 3

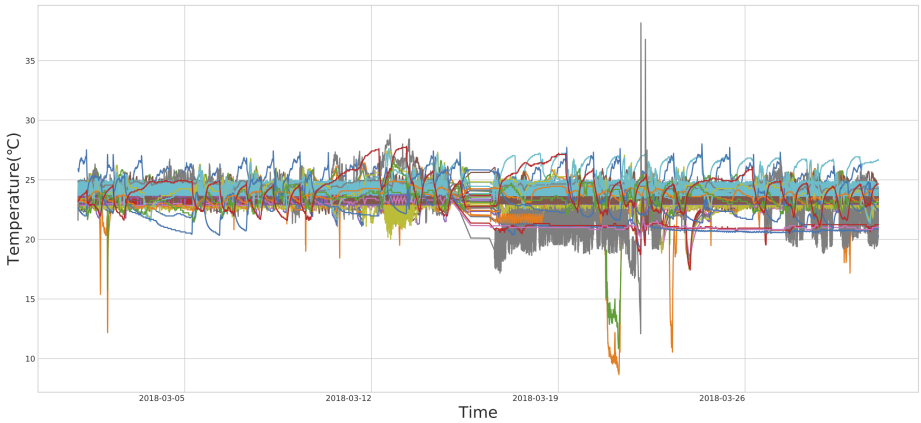

Figure 27: K-Medoids Cluster 4

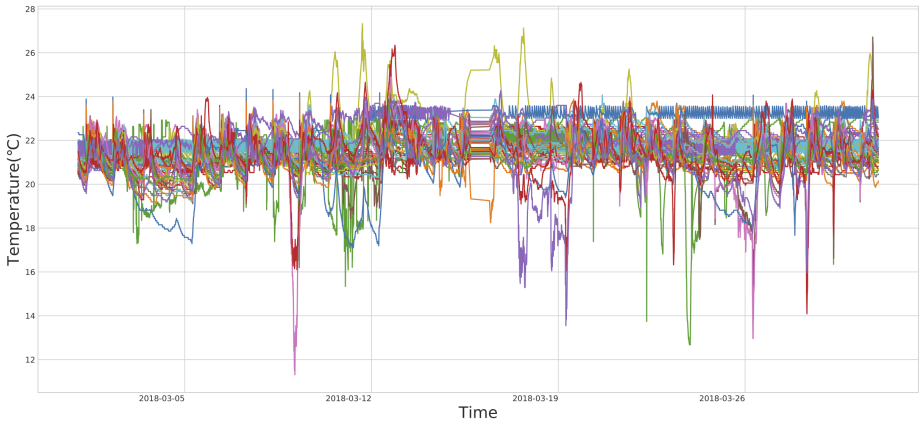

Figure 28: K-Medoids Cluster 5

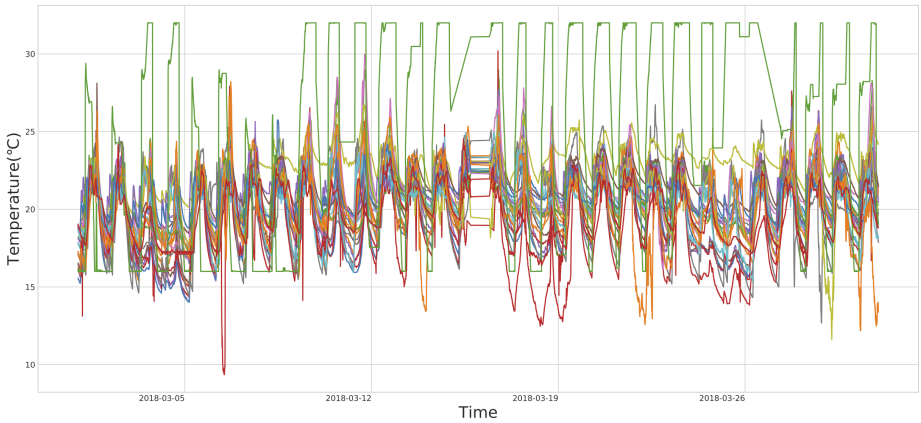

Figure 29: K-Medoids Cluster 6

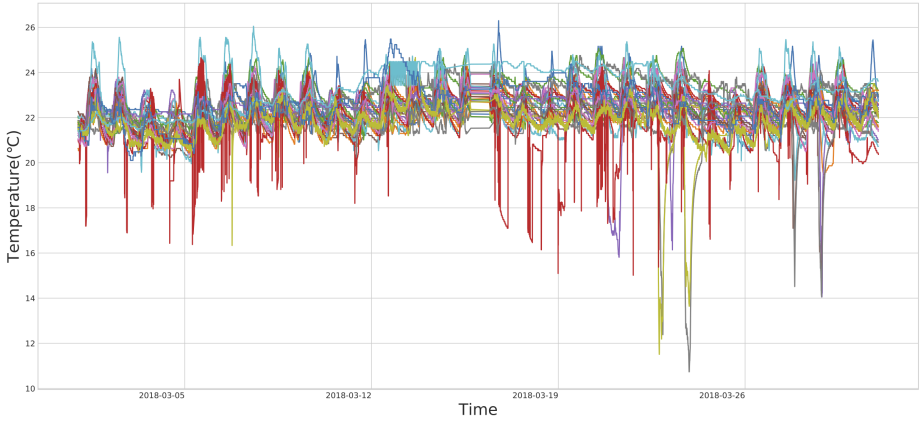

Figure 30: K-Medoids Cluster 7

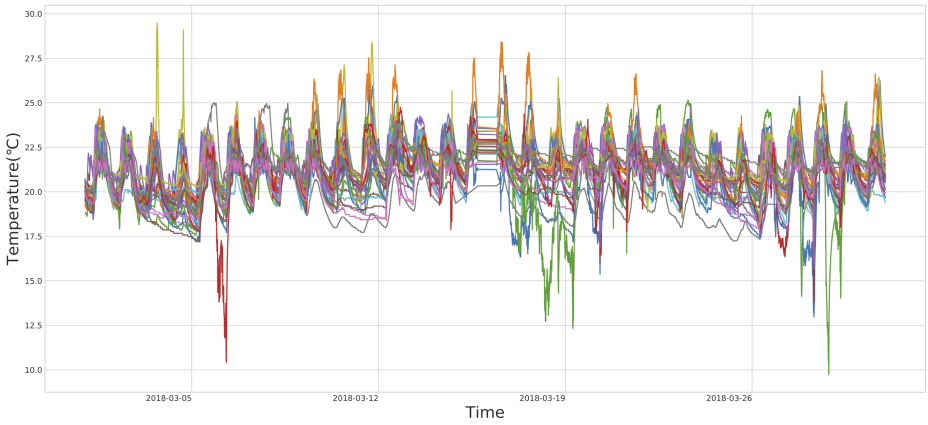

Figure 31: K-Medoids Cluster 8

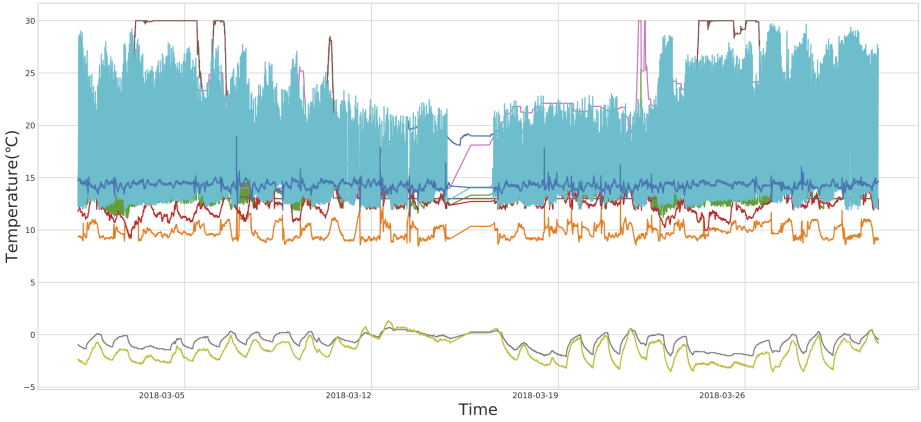

Figure 32: K-Medoids Cluster 9

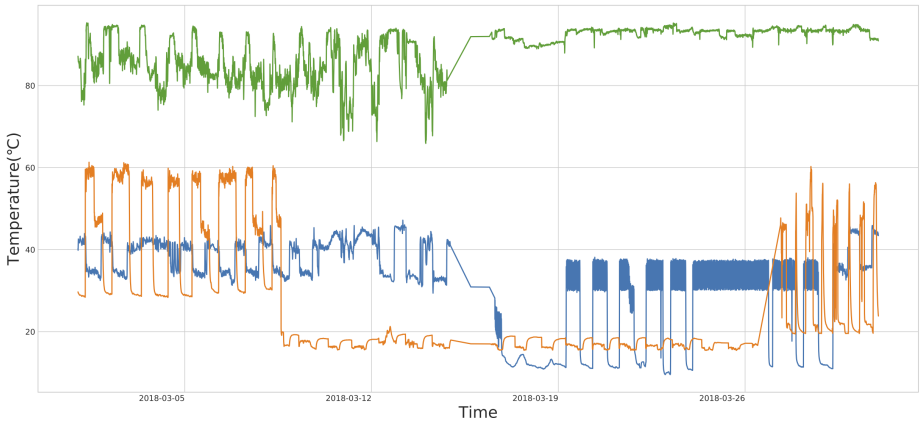

Figure 33: K-Medoids Cluster 10

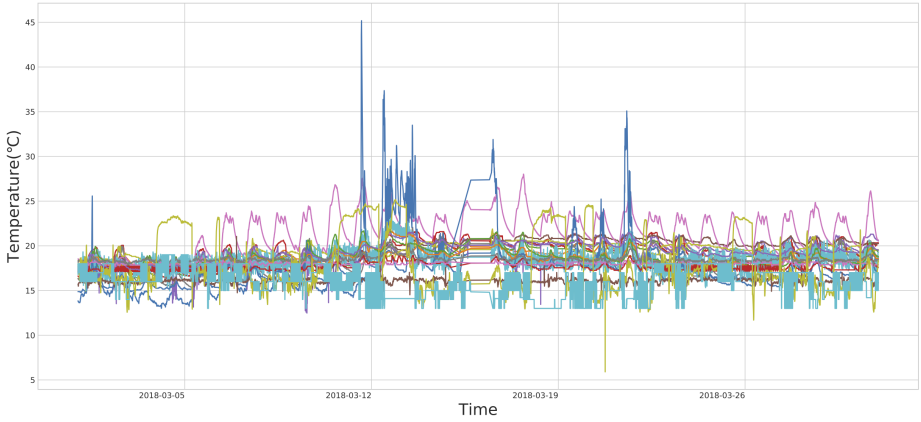

Figure 34: K-Medoids Cluster 11

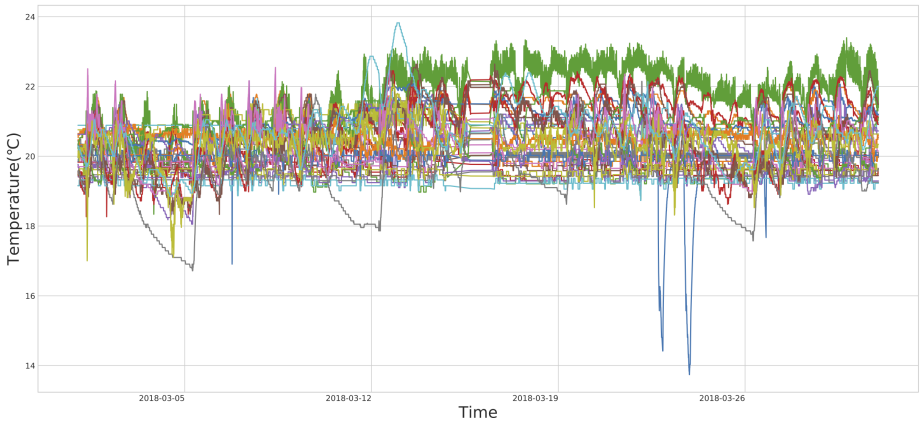

Figure 35: K-Medoids Cluster 12

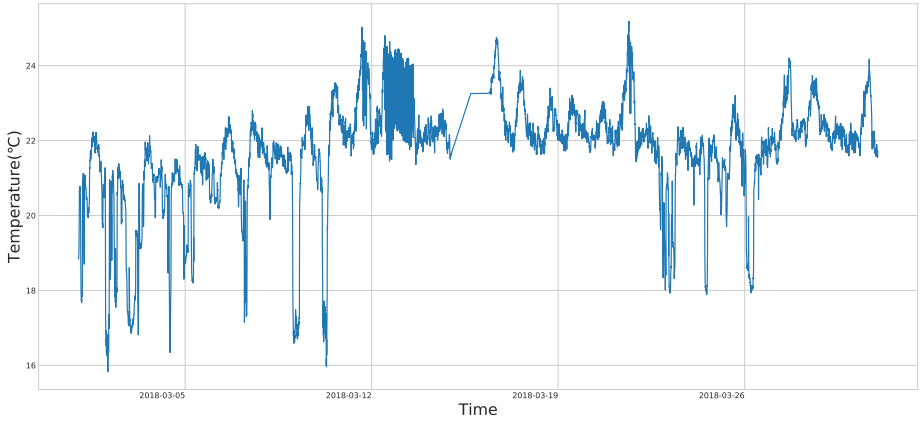

Figure 36: K-Medoids Cluster 13

3.4 Conclusions for different approaches

The test results from the previous subsections lead us to the following conclusions:

1. Dimensionality Reduction and DBSCAN are not ideal for long time trend log clustering.
2. Although K-Medoids can cluster long time trend logs, unstable results and high computational cost are the significant weaknesses.
3. Grouping trend logs by device type prior to clustering is a better choice for to reasons: a) the number of all logs is too large for efficient procesing, and b) the clusters among all the data seem to mainly distinguish device types, but not individual log behavior within a device.
4. When comparing the different methods considered here, K-means with Normalized Euclidean Distance appears to be the best option.

Note that temperature data with other sampling interval times have also be tested in this project. However, the results are excluded from discussion because they are largely similar to the 300 seconds sample interval temperature data.

4 Clustering result for RT and SAT Trend Logs

In this section, clustering results (K-Means) for Room Temperature (4.1) and Supply Air Temperature (4.2) with different sampling intervals are presented. To be specific, the ETL process is similar to the last section, which will not be detailed discussed in this section. Also, an interactive visualization (4.2.3) tool is shown in the last subsection, which helps to find the top N most similar trend logs and facilitate further exploration.

4.1 Room Temperature

In the subsection, Room Temperature Data with 300 seconds (4.1.1), 900 seconds (4.1.2), 1500 seconds (4.1.3) and 1800 (4.1.4) seconds sample time are presented.

4.1.1 Room Temperature with Sampling Interval 300 seconds

Through multiple tests, the chosen time range of the testing data is from November 15, 2018, to December 14, 2017, which has the minimum number of missing data and missing trend logs. Compared to the clustering results in the last section, the results of clustering are more evenly distributed and similar trend logs appear clustered in the same cluster.

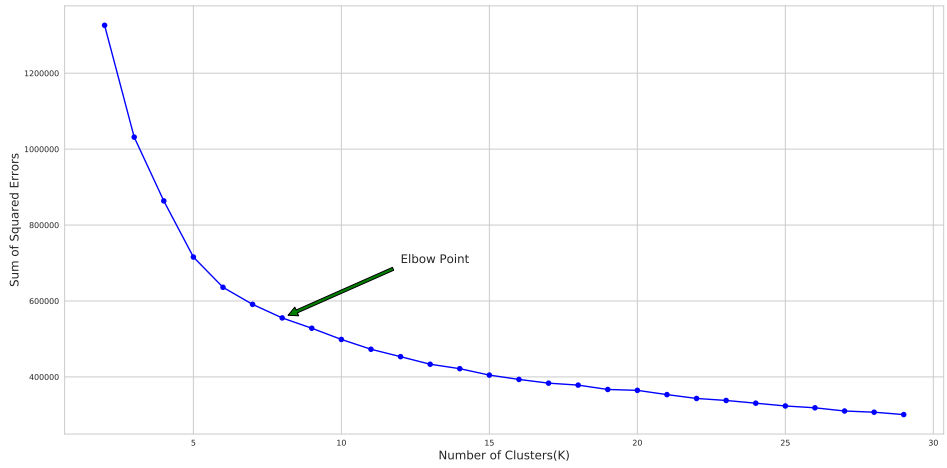

Figure 37: Sum of Squares Errors (RT 300 seconds interval)

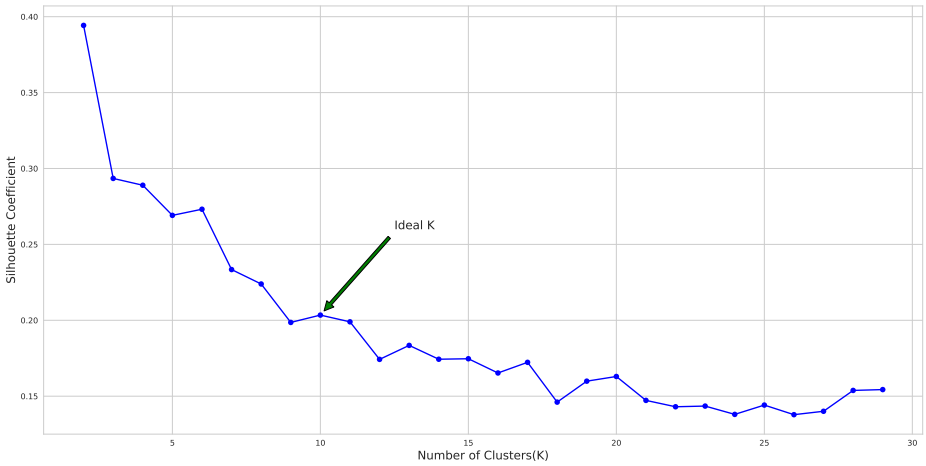

Figure 38: Silhouette Coefficient (RT 300 seconds interval)

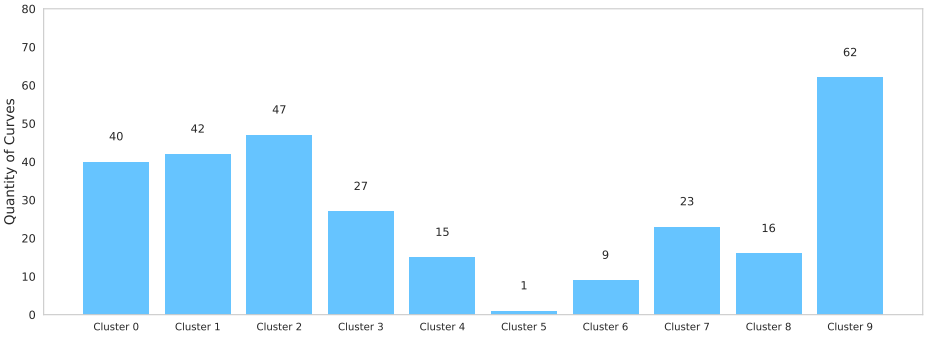

Figure 39: Cluster Distribution (RT 300 seconds interval)

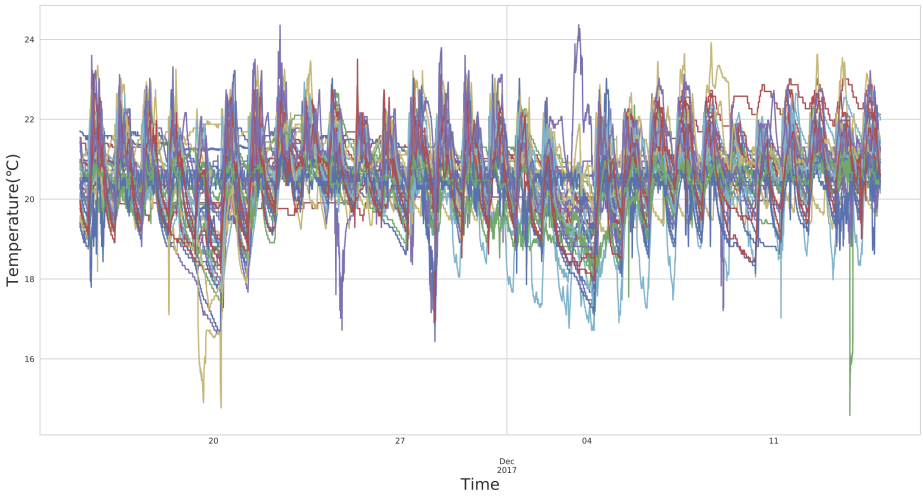

Figure 40: Cluster 0 (RT 300 seconds interval)

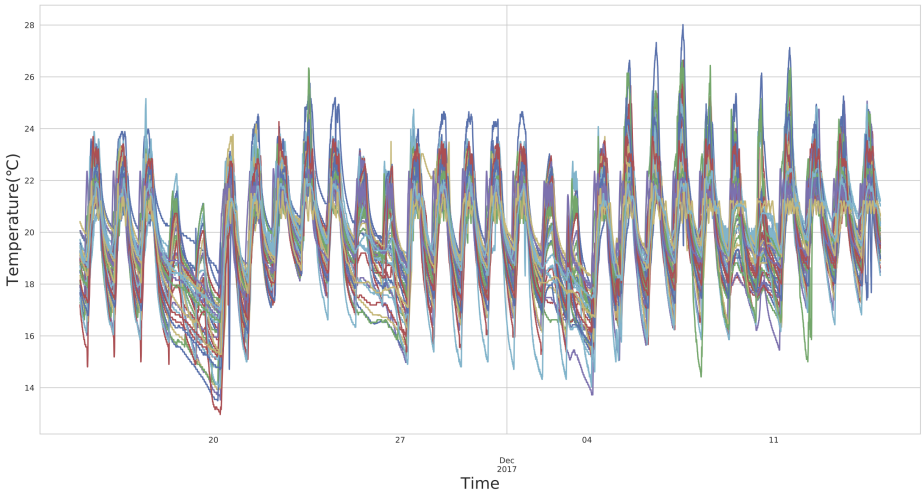

Figure 41: Cluster 1 (RT 300 seconds interval)

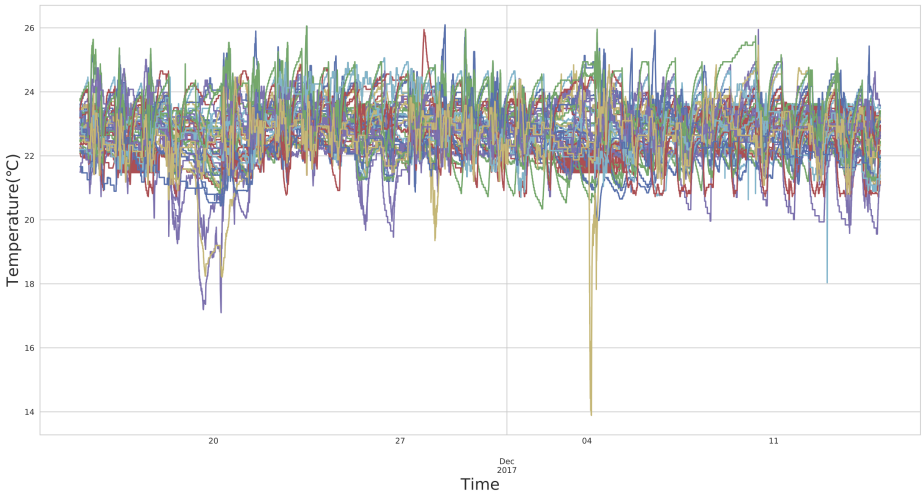

Figure 42: Cluster 2 (RT 300 seconds interval)

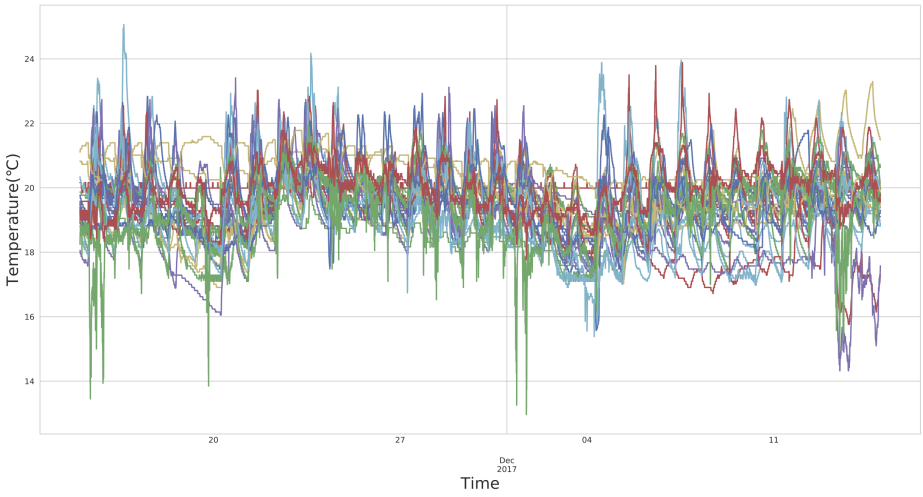

Figure 43: Cluster 3 (RT 300 seconds interval)

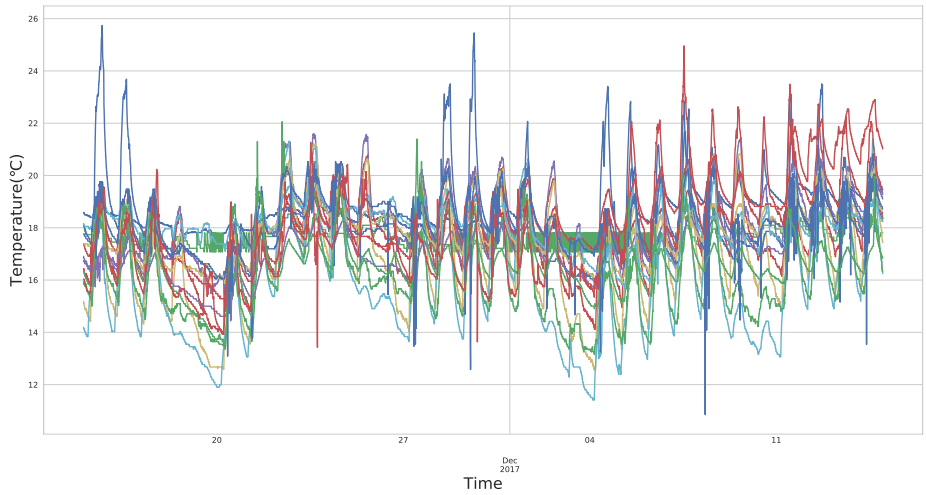

Figure 44: Cluster 4 (RT 300 seconds interval)

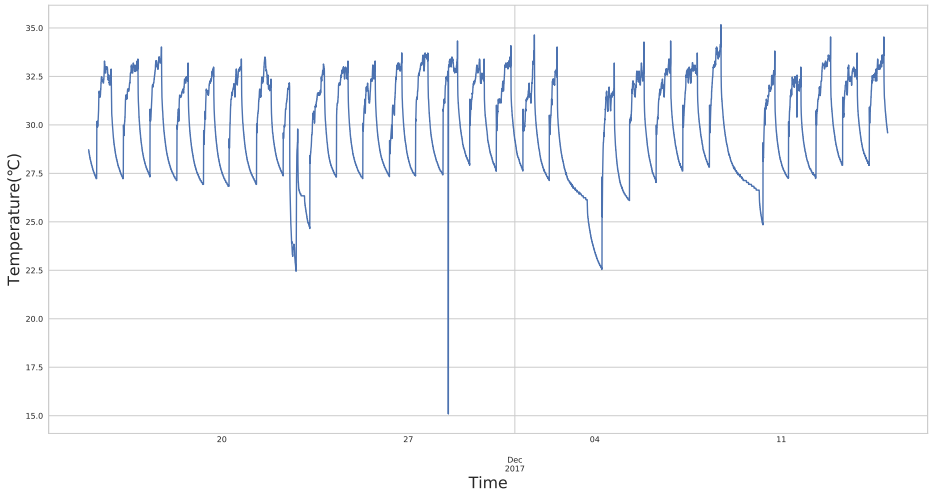

Figure 45: Cluster 5 (RT 300 seconds interval)

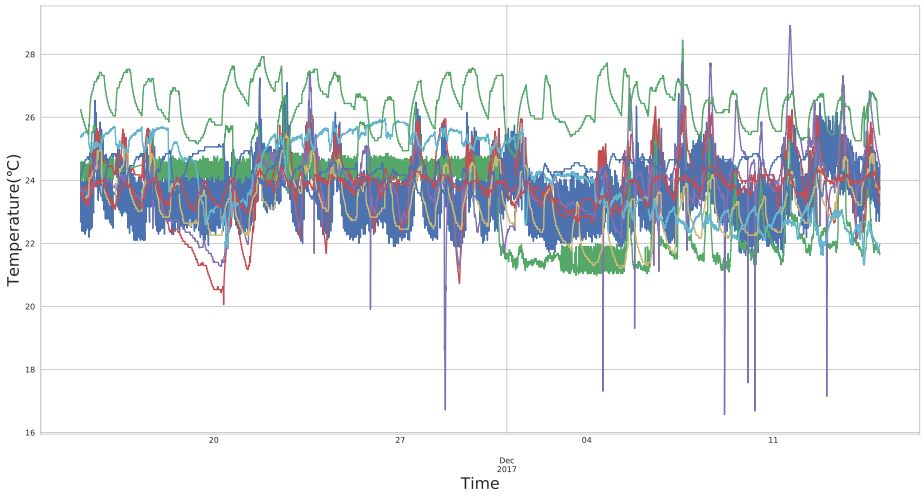

Figure 46: Cluster 6 (RT 300 seconds interval)

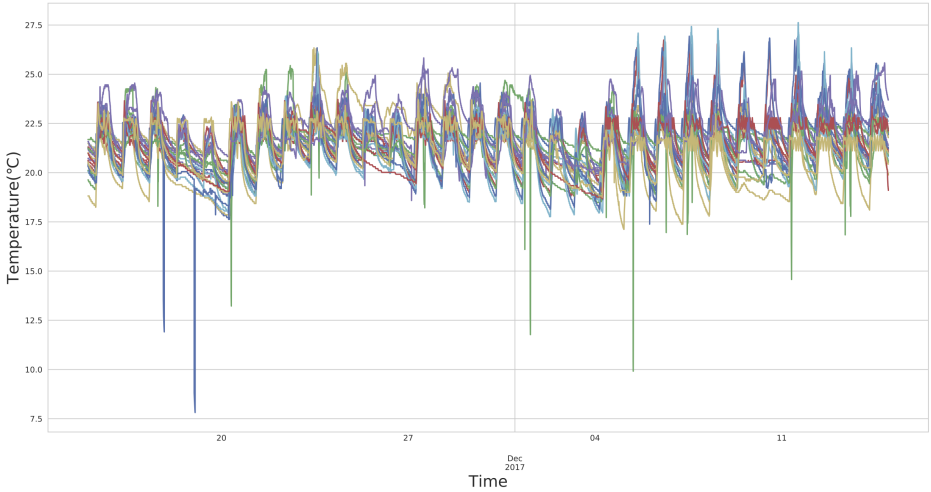

Figure 47: Cluster 7 (RT 300 seconds interval)

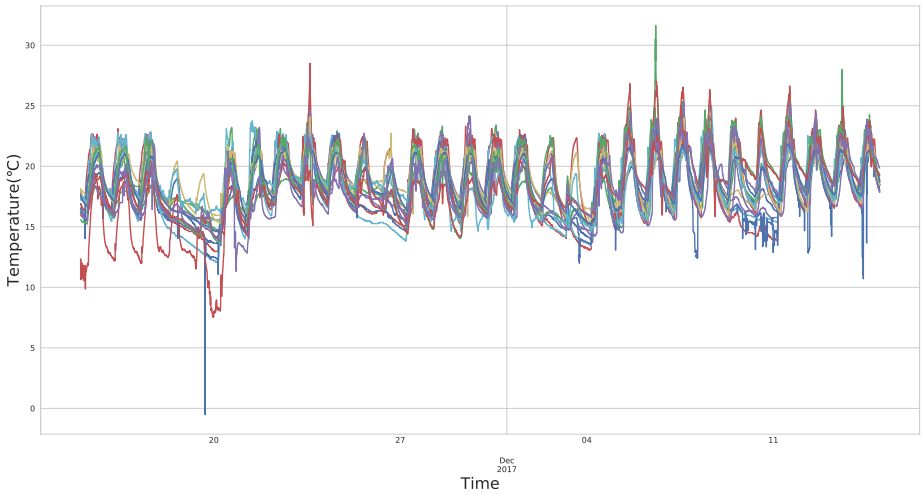

Figure 48: Cluster 8 (RT 300 seconds interval)

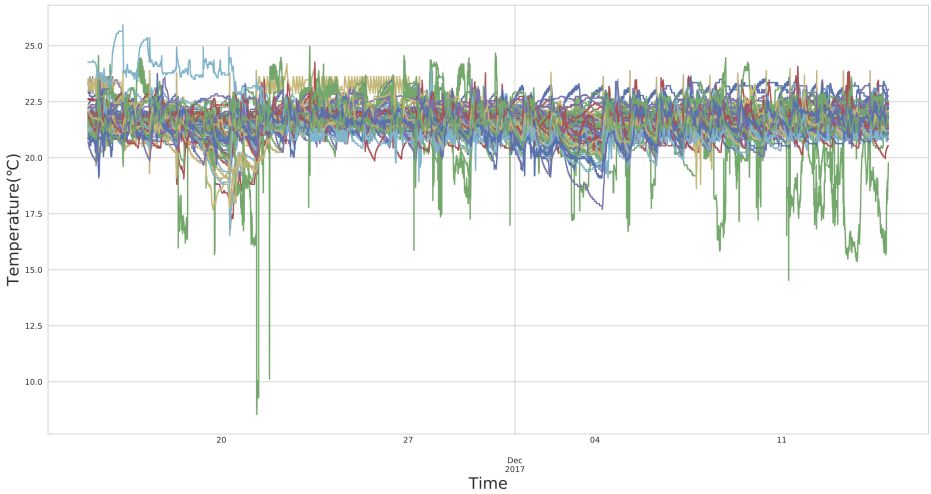

Figure 49: Cluster 9 (RT 300 seconds interval)

4.1.2 Room Temperature with Sampling Interval 900 seconds

The clustering results for RT with a sampling interval of 900 seconds are not as good as RT with the sampling interval of 300 seconds. The main reason for that is missing data, supposedly since some of the trend logs are link noise data.

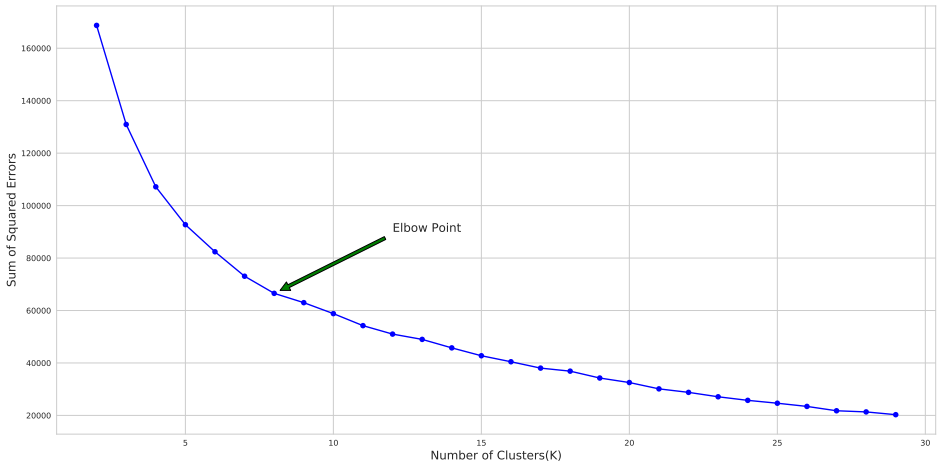

Figure 50: Sum of Squares Errors (RT 900 seconds interval)

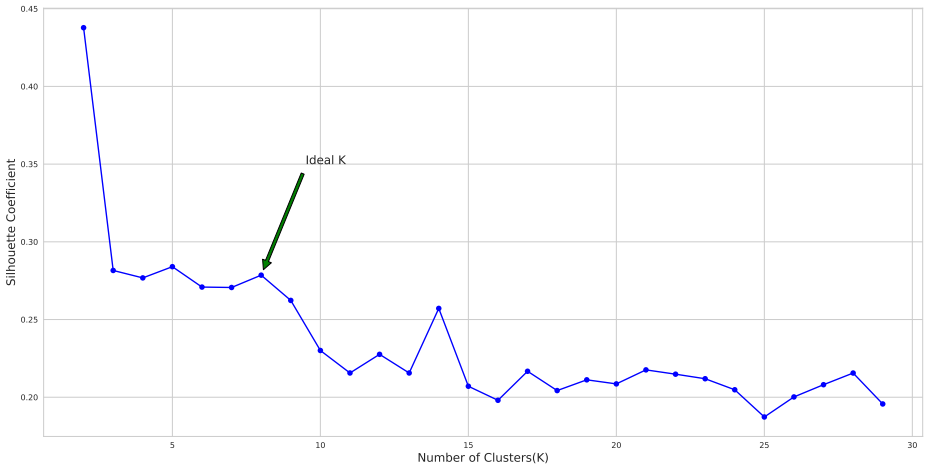

Figure 51: Silhouette Coefficient (RT 900 seconds interval)

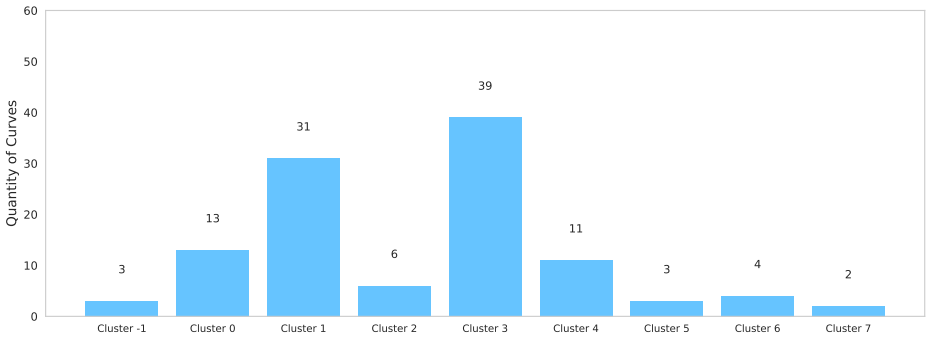

Figure 52: Clusters Distribution (RT 900 seconds interval)

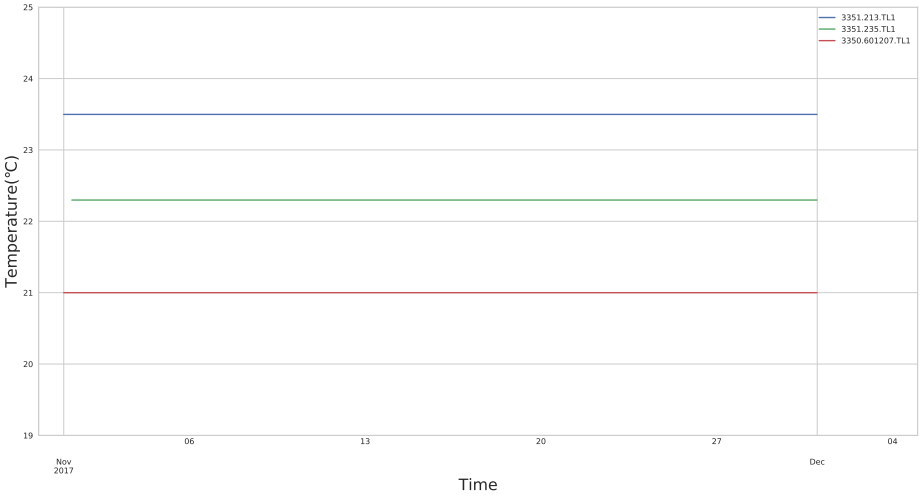

Figure 53: Cluster -1 (RT 900 seconds interval)

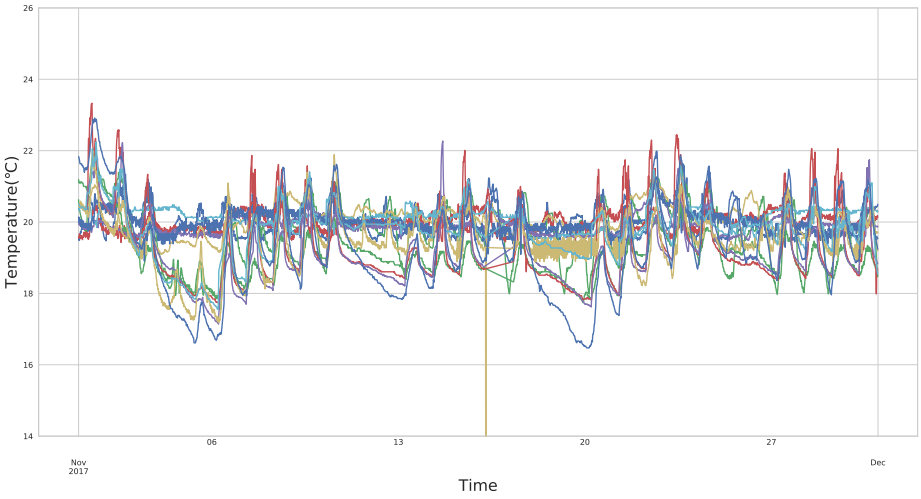

Figure 54: Cluster 0 (RT 900 seconds interval)

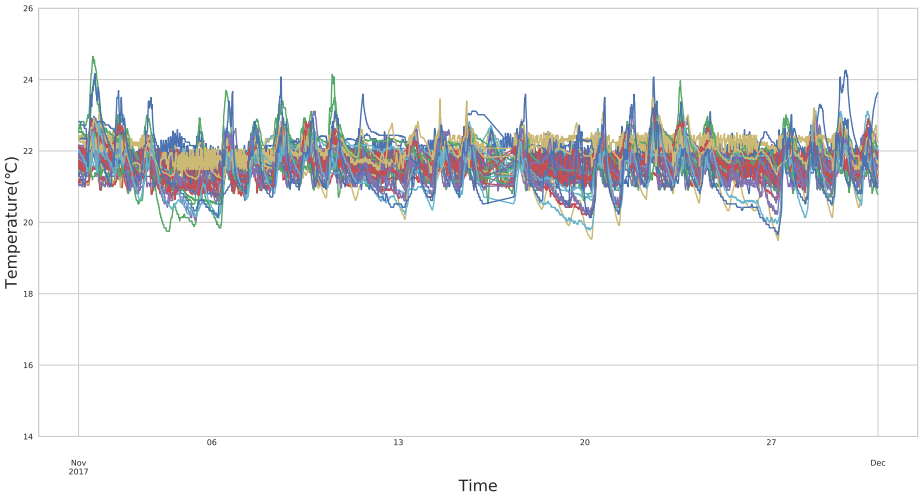

Figure 55: Cluster 1 (RT 900 seconds interval)

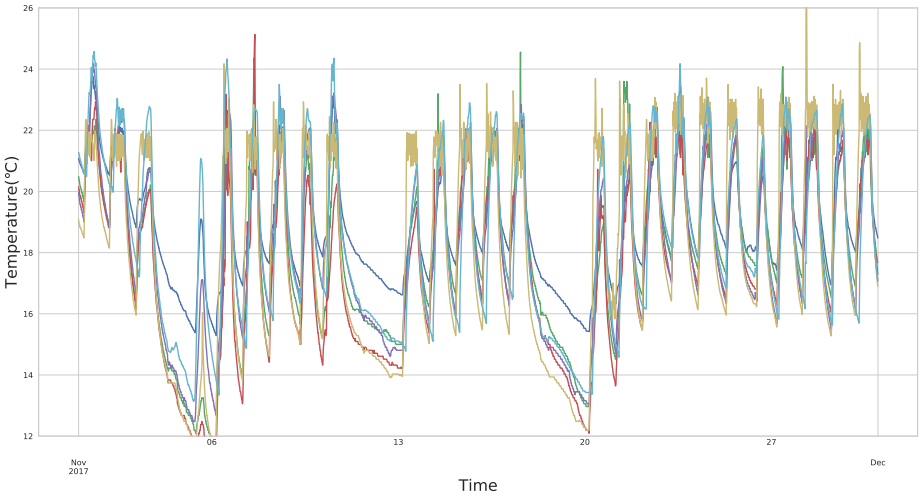

Figure 56: Cluster 2 (RT 900 seconds interval)

Figure 57: Cluster 3 (RT 900 seconds interval)

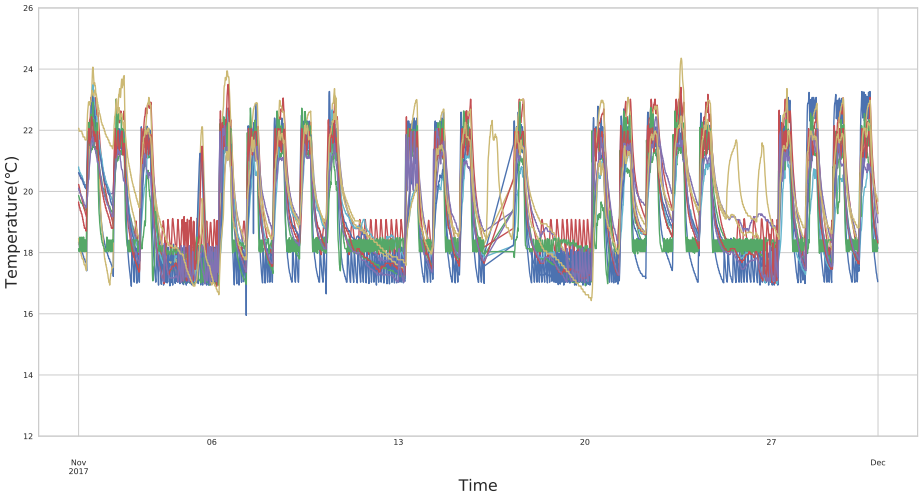

Figure 58: Cluster 4 (RT 900 seconds interval)

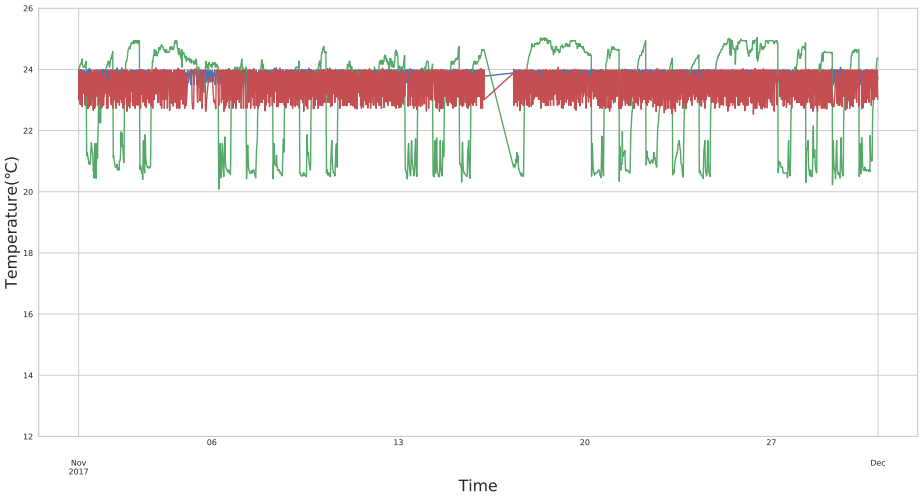

Figure 59: Cluster 5 (RT 900 seconds interval)

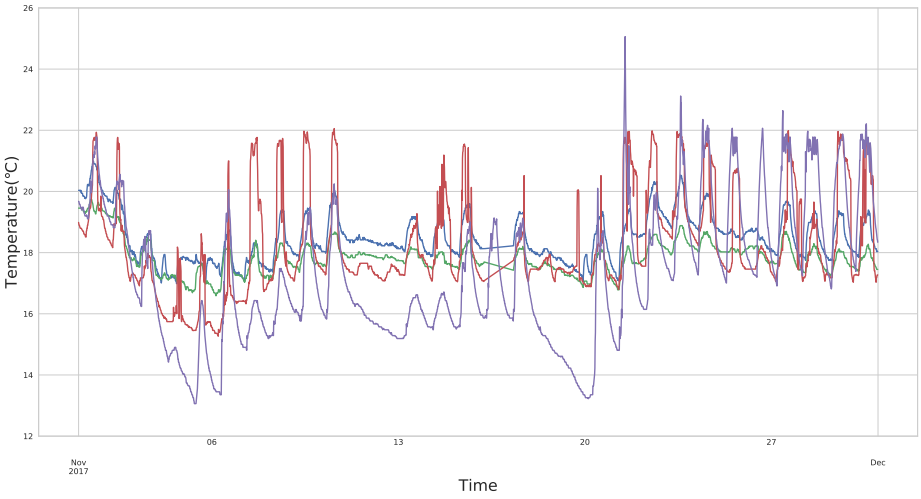

Figure 60: Cluster 6 (RT 900 seconds interval)

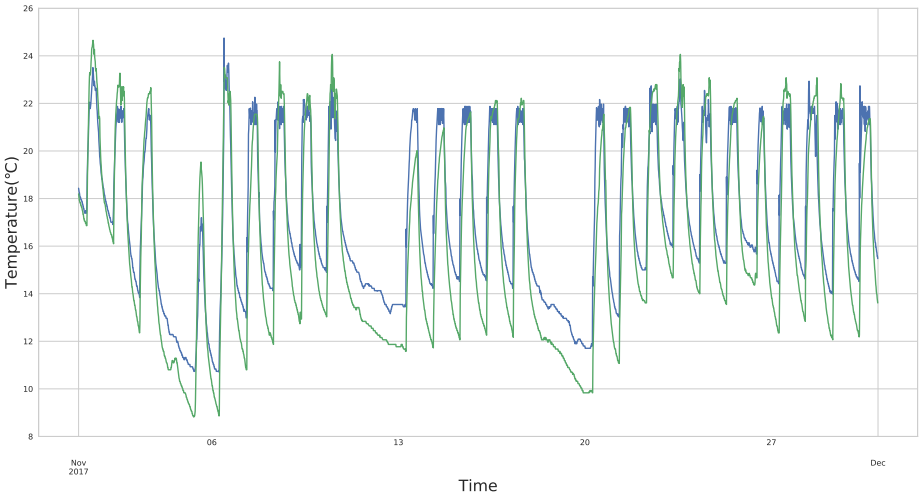

Figure 61: Cluster 7 (RT 900 seconds interval)

4.1.3 Room Temperature with Sampling Interval 1500 seconds

Compared to the previous two types of RT, the number of RT with a sampling Interval of 1500 seconds is quite small. From the figures below you can see, almost all trends logs in cluster 1 are the same in variation trend and other trend logs seem cluster by different temperature variation range.

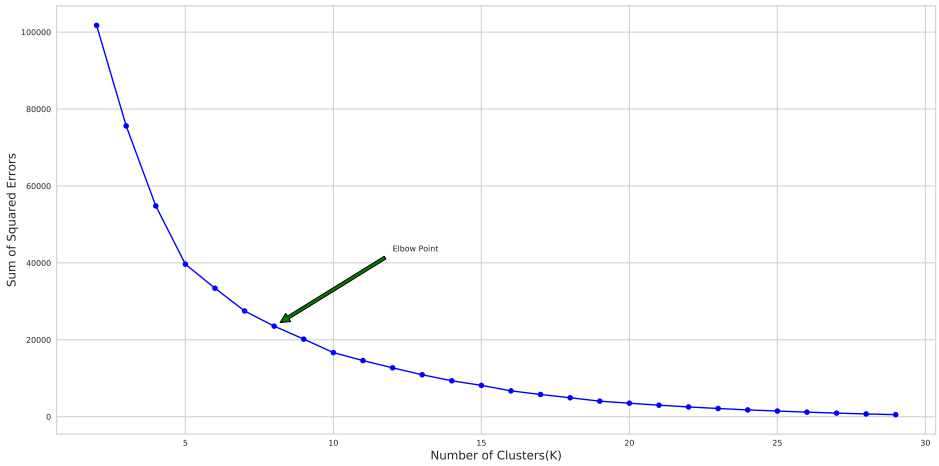

Figure 62: Sum of Square Errors (RT 1500 seconds interval)

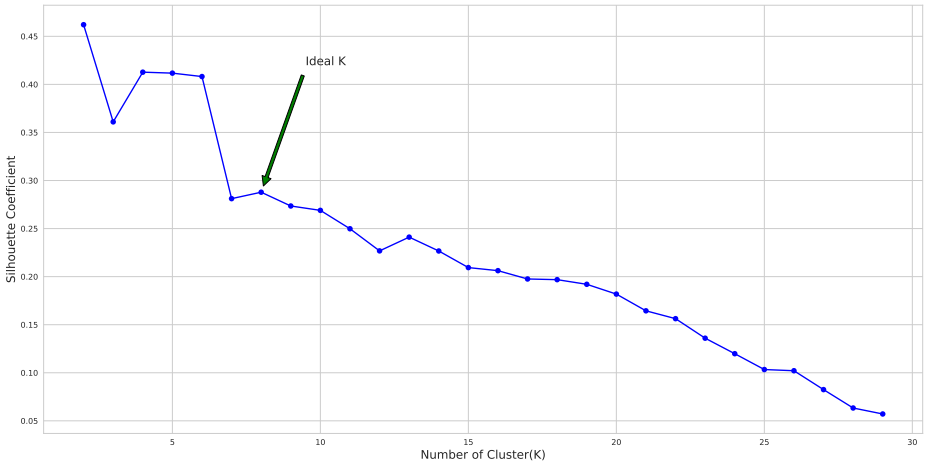

Figure 63: Silhouette Coefficient (RT 1500 seconds interval)

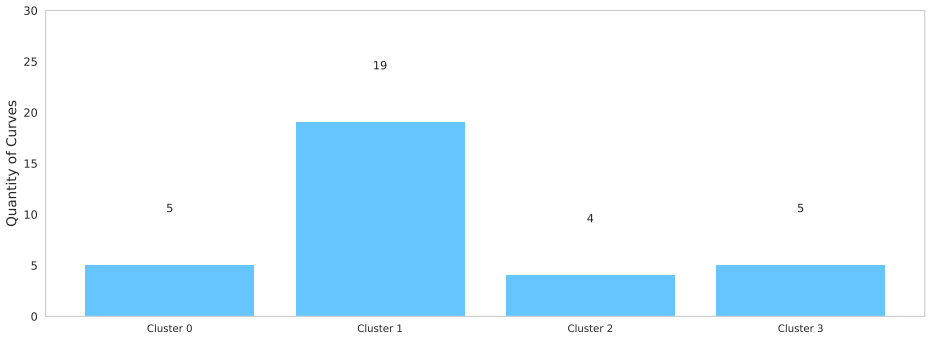

Figure 64: Clusters Distribution (RT 1500 seconds interval)

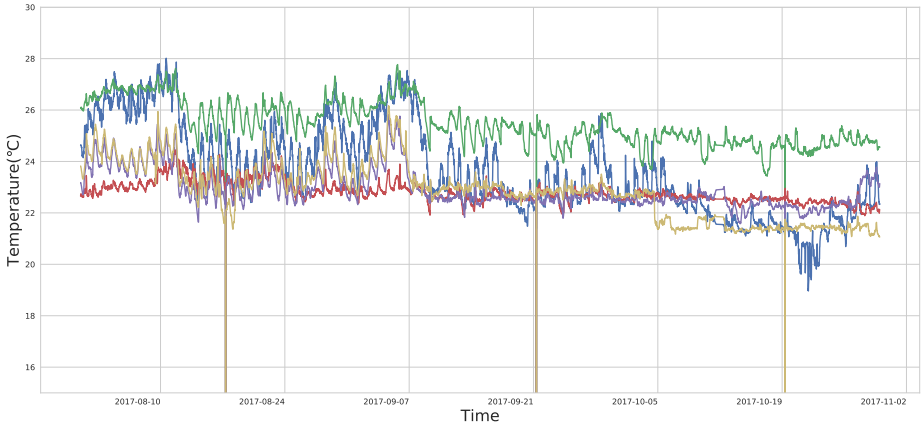

Figure 65: Cluster 0 (RT 1500 seconds interval)

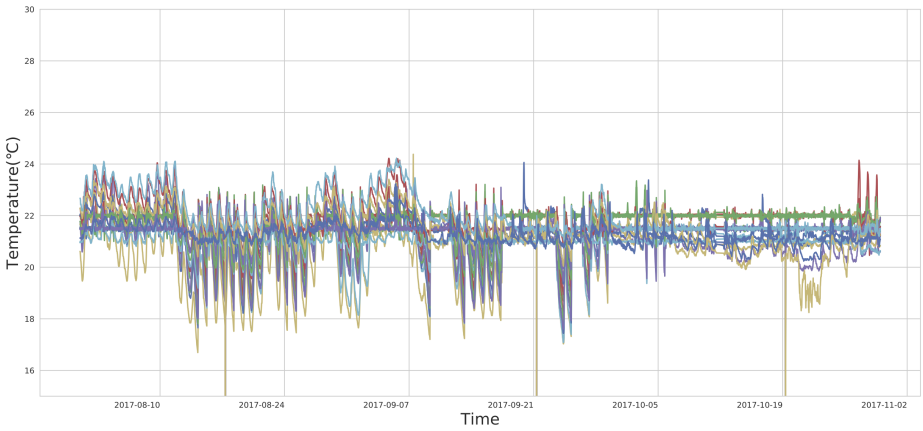

Figure 66: Cluster 1 (RT 1500 seconds interval)

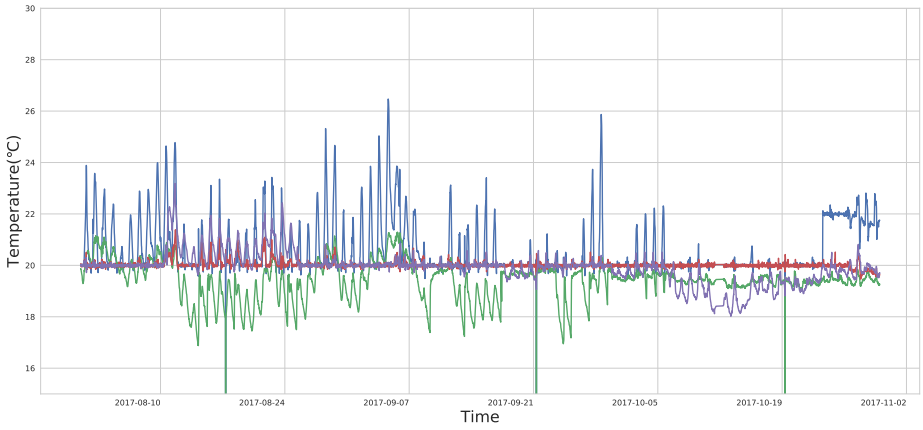

Figure 67: Cluster 2 (RT 1500 seconds interval)

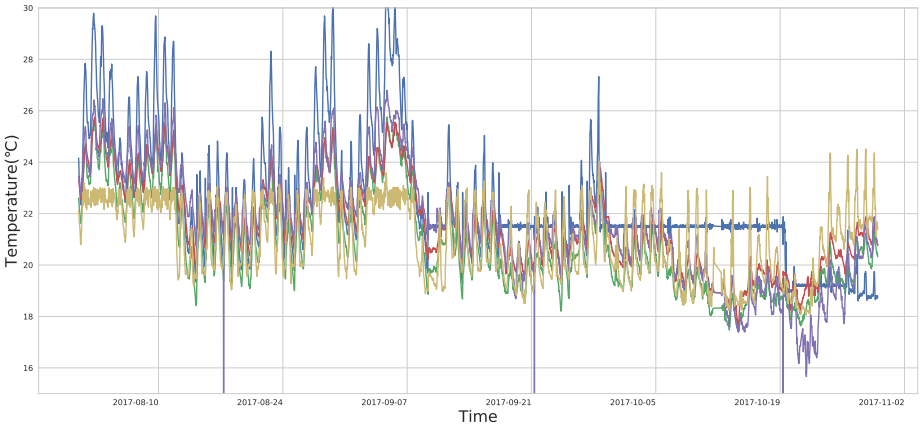

Figure 68: Cluster 3 (RT 1500 seconds interval)

4.1.4 Room Temperature with Sampling Interval 1800 seconds

Since there are only two samples in RT with sampling interval 1800 seconds and they are almost just the same. The below figure shows these two trend logs.

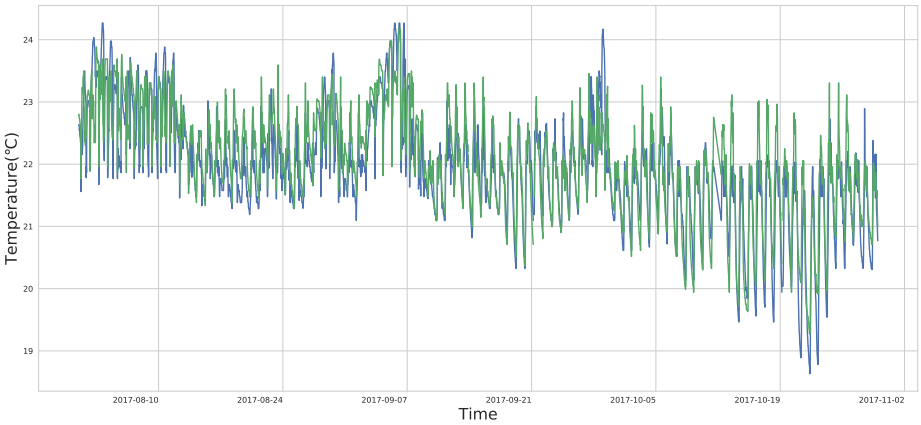

Figure 69: (RT 1500 seconds interval trend logs)

4.2 Supply Air Temperature

Supply Air Temperature trend logs with 300 seconds (4.2.1) and 900 seconds (4.2.2) sample time are presented in this section.

4.2.1 Supply Air Temperature with Sampling Interval 300 seconds

The clustering results for SAT with sampling interval are not as good as the results from RAT since there are too many missing data. Among them, there is no particularly similar trend log which can be seen in the following figures.

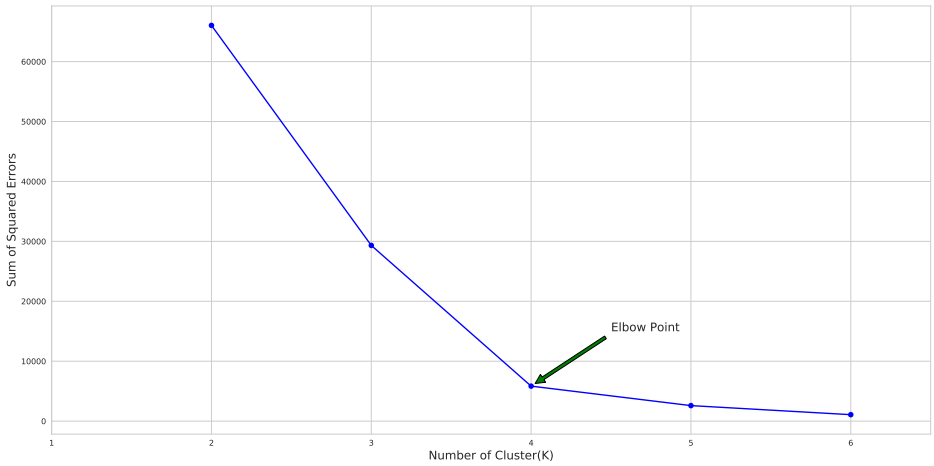

Figure 70: Sum of Square Errors (SAT 300 seconds interval)

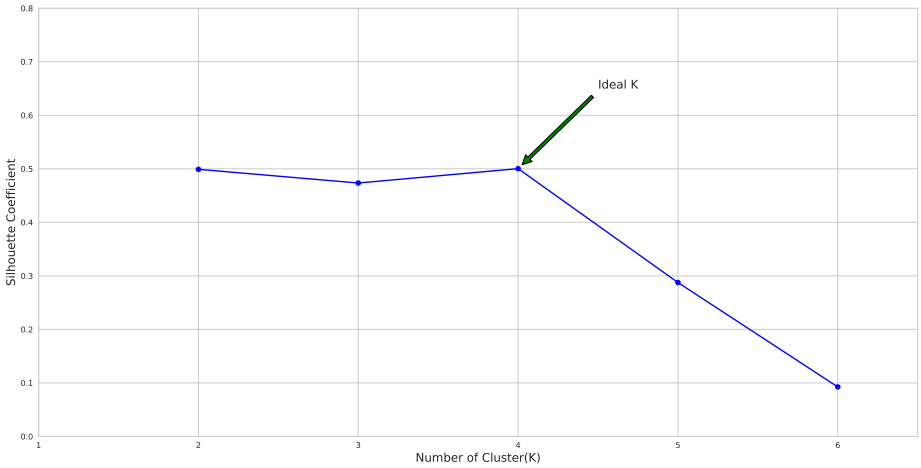

Figure 71: Silhouette Coefficient (SAT 300 seconds interval)

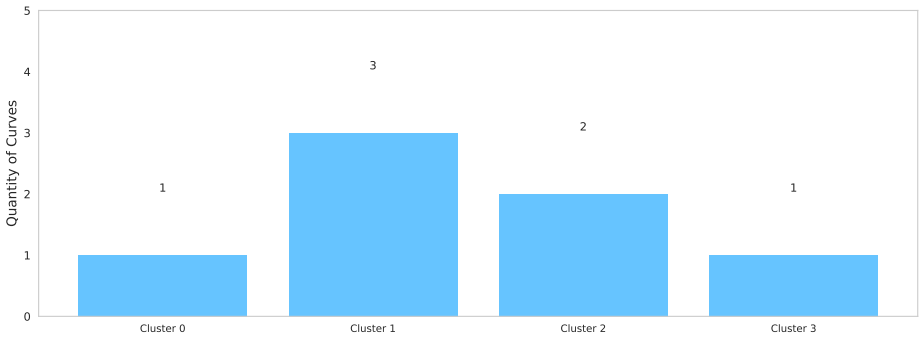

Figure 72: Clusters Distribution (SAT 300 seconds interval)

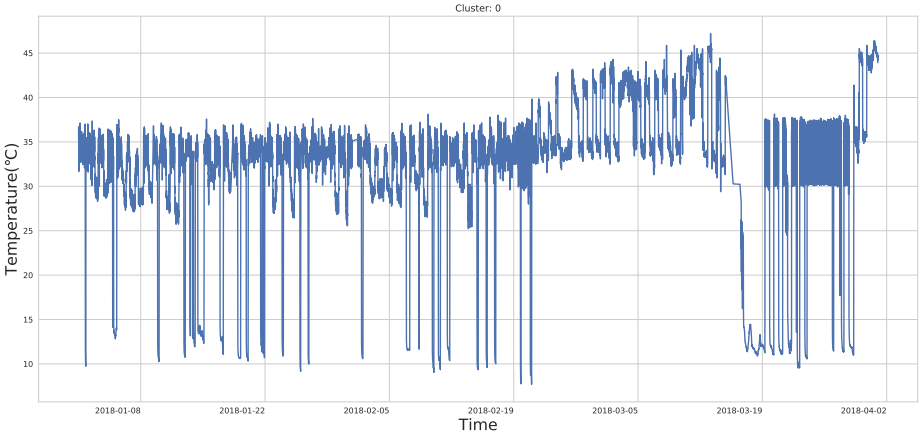

Figure 73: Cluster 0 (SAT 300 seconds interval)

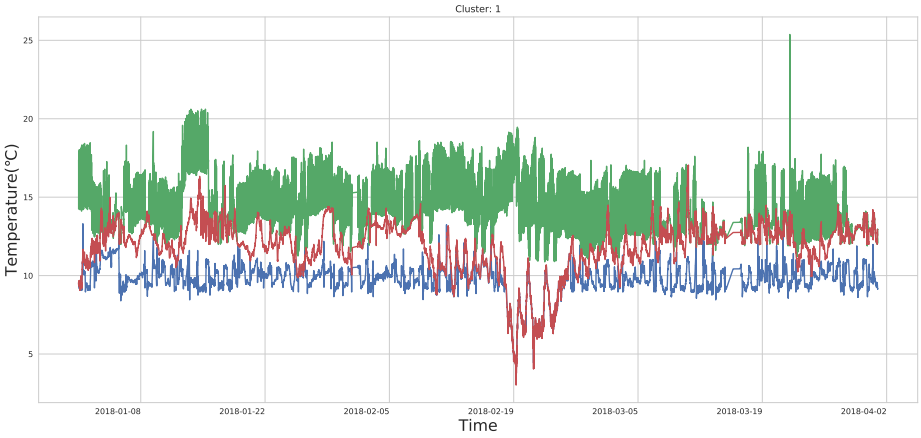

Figure 74: Cluster 1 (SAT 300 seconds interval)

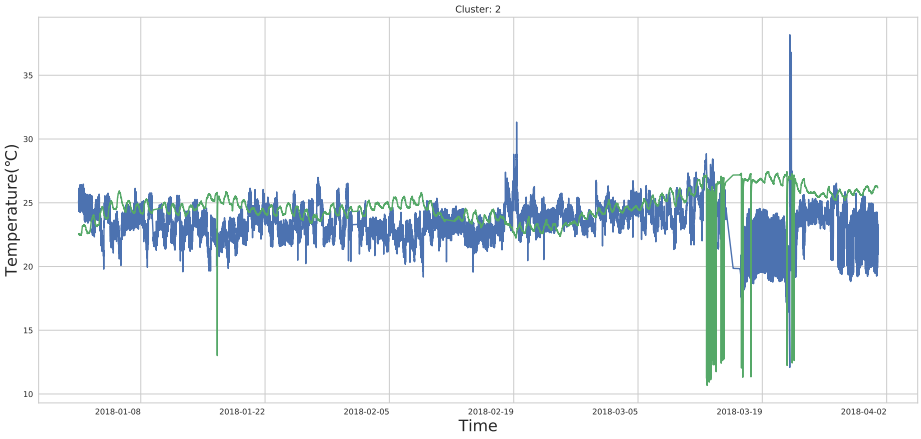

Figure 75: Cluster 2 (SAT 300 seconds interval)

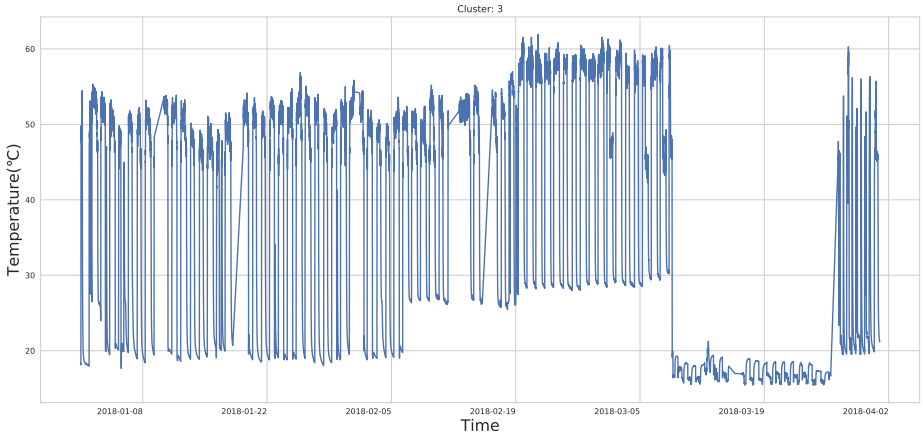

Figure 76: Cluster 3 (SAT 300 seconds interval)

4.2.2 Supply Air Temperature with Sampling Interval 900 seconds

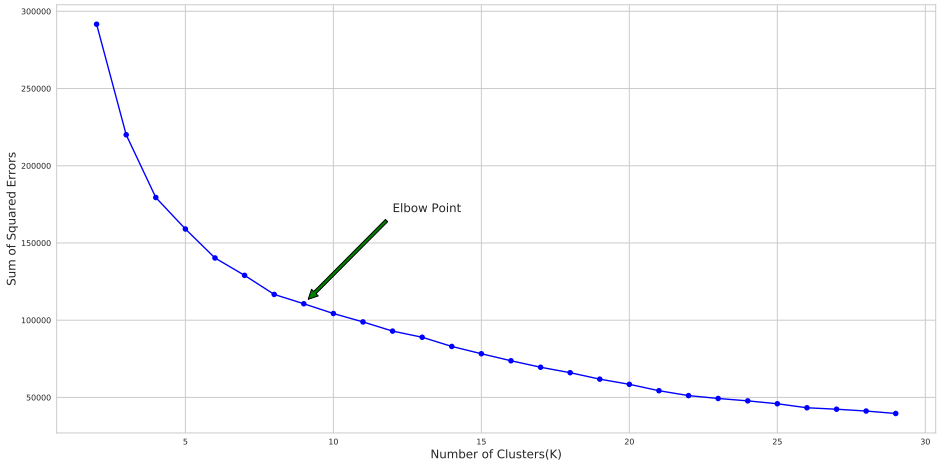

Figure 77: Sum of Square Errors (SAT 900 seconds interval)

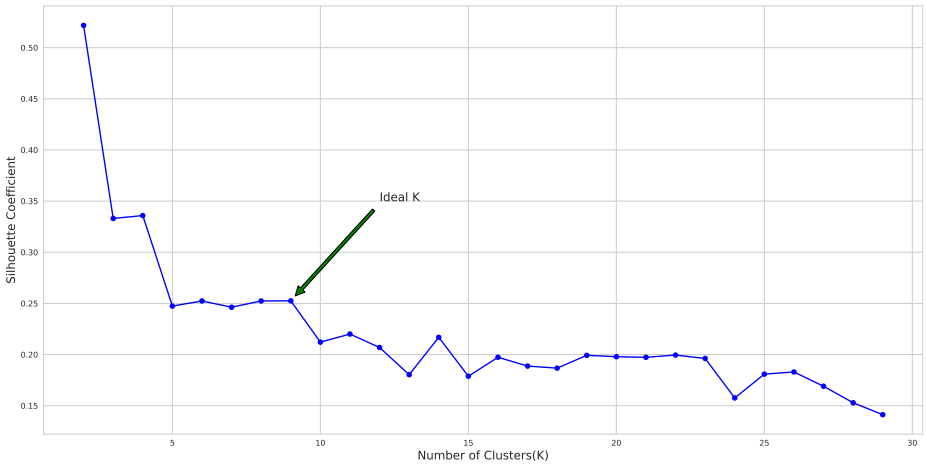

Figure 78: Silhouette Coefficient (SAT 900 seconds interval)

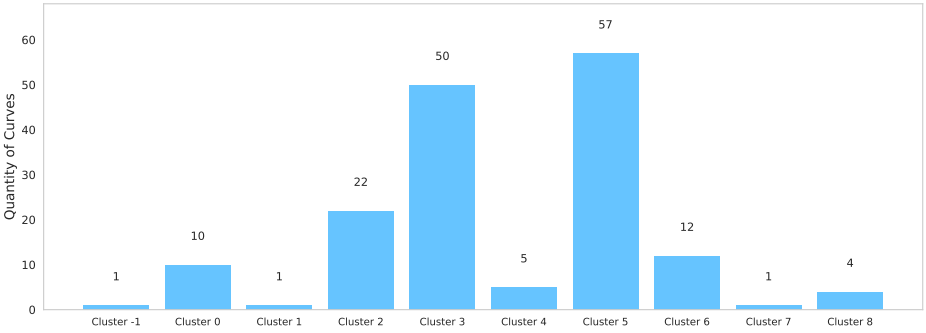

Figure 79: Clusters Distribution (SAT 900 seconds interval)

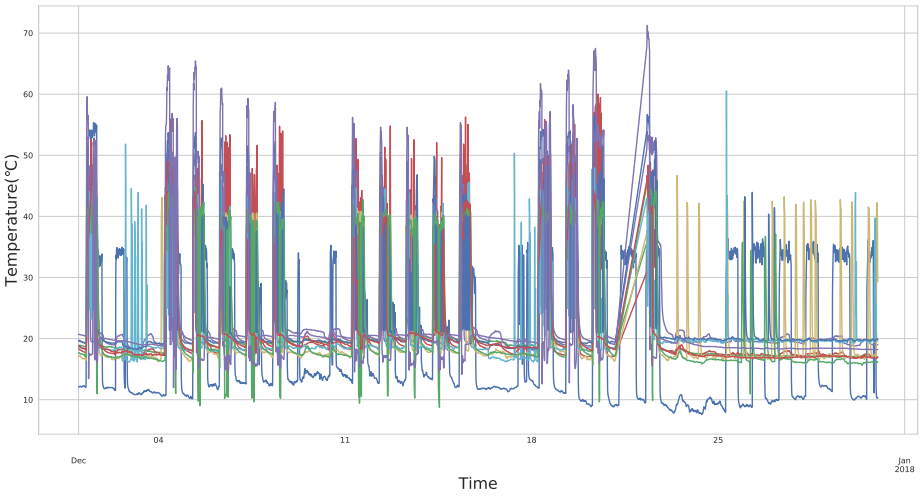

Figure 80: Cluster 0 (SAT 900 seconds interval)

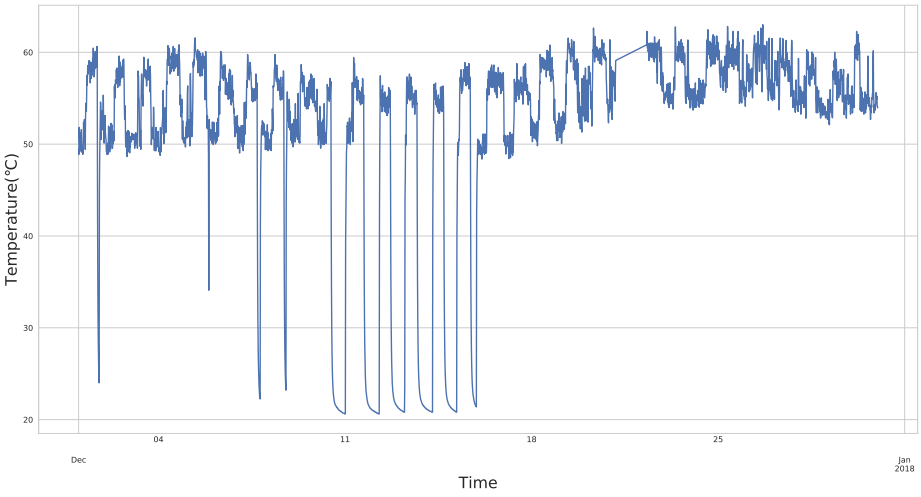

Figure 81: Cluster 1 (SAT 900 seconds interval)

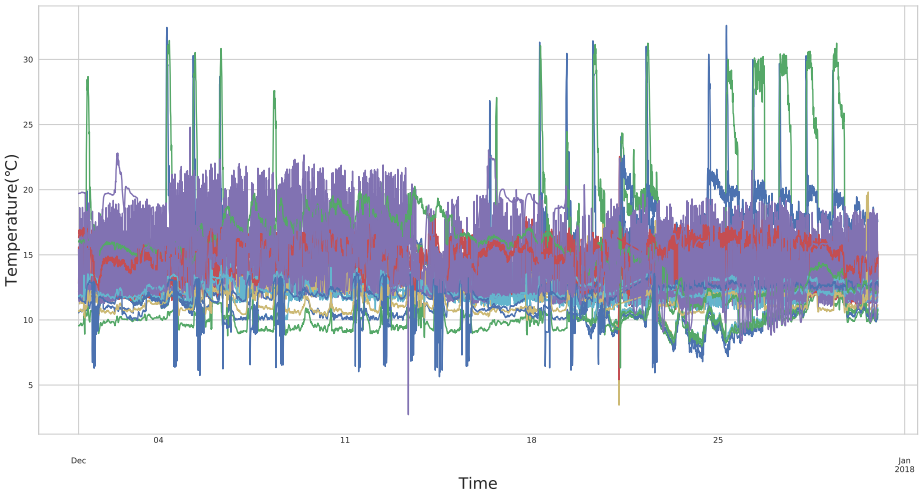

Figure 82: Cluster 2 (SAT 900 seconds interval)

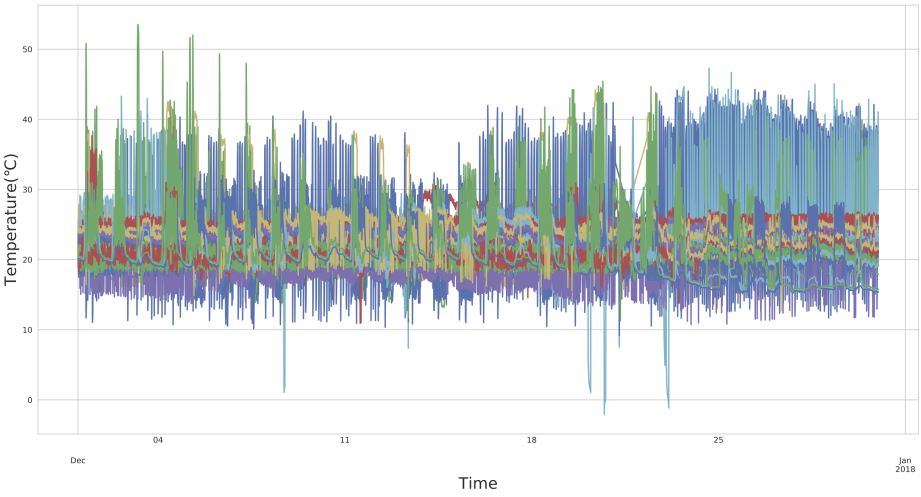

Figure 83: Cluster 3 (SAT 900 seconds interval)

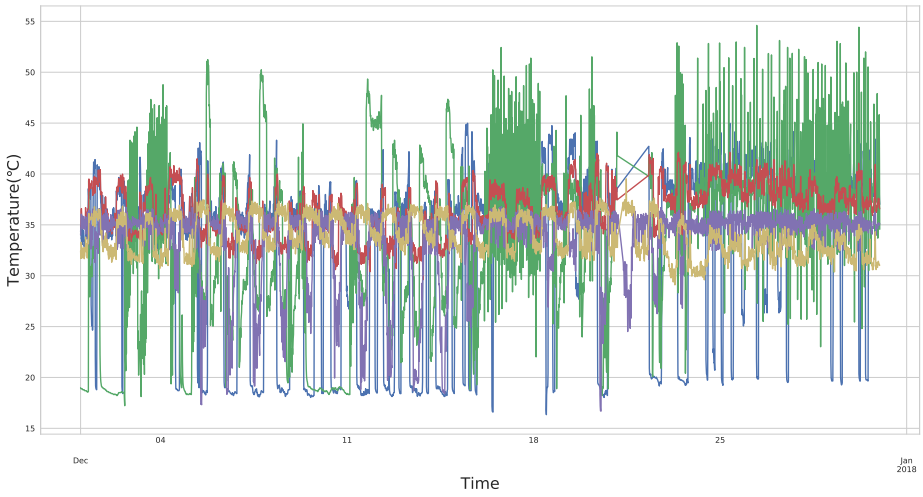

Figure 84: Cluster 4 (SAT 900 seconds interval)

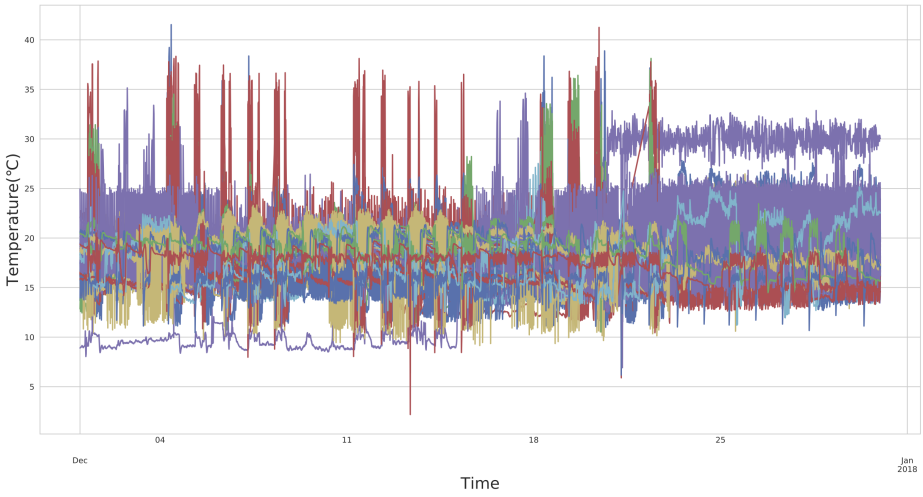

Figure 85: Cluster 5 (SAT 900 seconds interval)

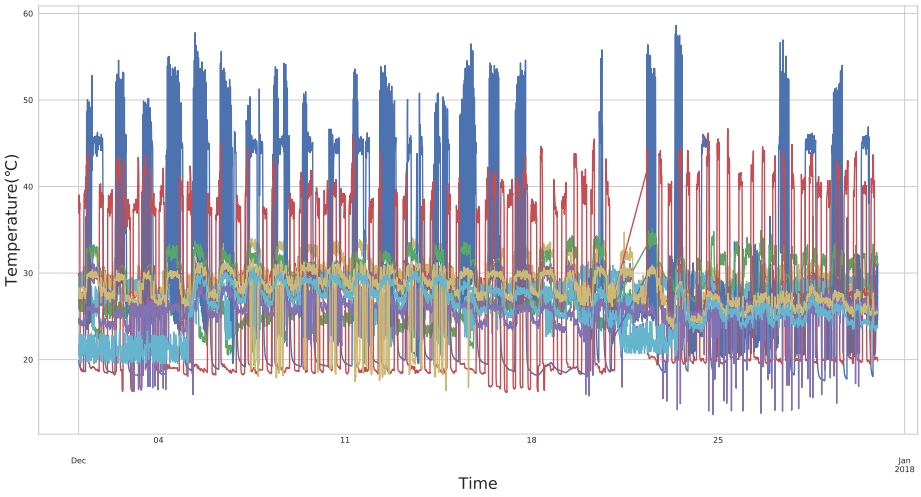

Figure 86: Cluster 6 (SAT 900 seconds interval)

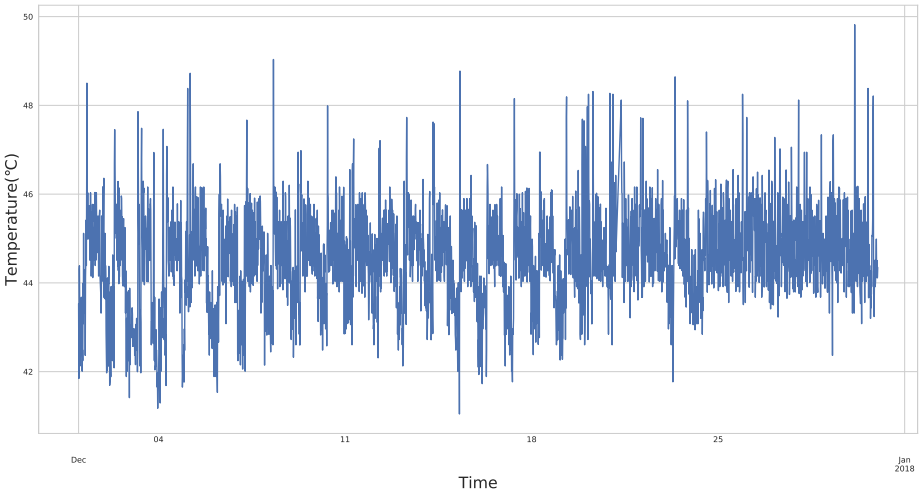

Figure 87: Cluster 7 (SAT 900 seconds interval)

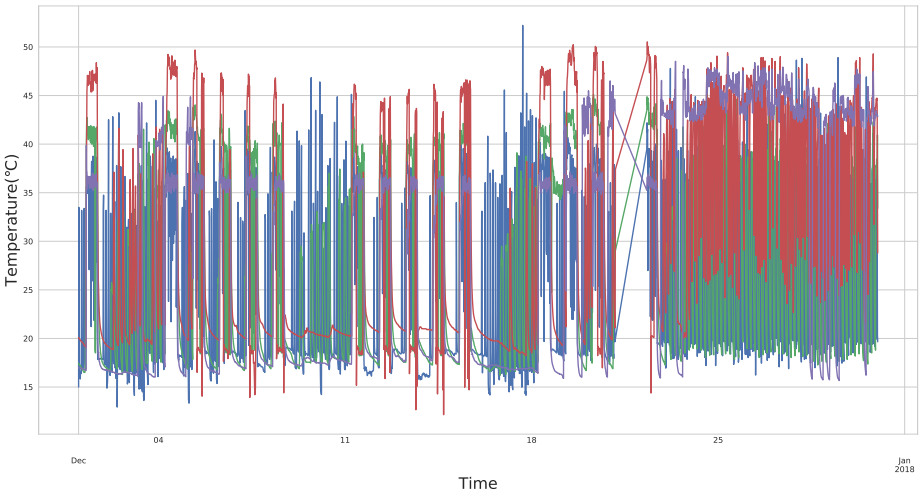

Figure 88: Cluster 8 (SAT 900 seconds interval)

4.2.3 Interactive Visualization for Exploration

In order to analyze the variation trend of a trend log, an interactive visualization demo is developed for interactive exploration. Currently, it only supports Room Temperature and Supply Air Temperature logs. The procedure to use it is as follows:

1. Choose the Temperature Type, which can be chosen from Room Temperature and Supply Air Temperature
2. Choose the Interval Time, i.e. the sampling time interval for your chosen temperature type
3. Choose the Cluster you want to explore
4. Choose the Curve Id you want to focus on
5. Choose the Similar Num, which means the top N nearest trend logs within the same cluster that you want to compare with the focus Curve Id
6. Choose Start Time and End Time, i.e. the time range of the data for display (not for clustering)
6. Click *Show the curves* to update the view

The user interface and the display results are shown in the following figure:

Figure 89: Selection Menu

The following 3 figures show the results from the previous selections. The first figure shows the N nearest trend logs for the chosen focus curve and the distance matrix, building info, trend log

name and their corresponding set point trend log names. The second figure displays these curves in the chosen time range. The last figure shows periodic curves of the chosen curve and its set point, which includes the previous and following periods, last year and the next year curves. In both of the last two figures curves can be hidden by clicking their labels in the legend.

	3351.102.TL28	SAT_SP_id	device_name	TL_name	SP_name	Build_id
3351.102.TL28	0.00000000000000	3351.102.TL33	Coil-818	TS2_Coil-818_SAT_POLL_TL	Coil-826_SAT_SP_POLL_TL	3351
3351.103.TL22	48.64589207639560	3351.103.TL26	Coil-802	TS2_Coil-802_SAT_POLL_TL	Coil-802_SAT_SP_POLL_TL	3351
3351.117.TL52	62.46055090636443	3351.117.TL55	Coil-611	TS2_Coil-611_SAT_POLL_TL	Coil-611_SAT_SP_POLL_TL	3351
3351.108.TL23	72.49195944329819	3351.108.TL27	Coil-716	TS2_Coil-716_SAT_POLL_TL	Coil-716_SAT_SP_POLL_TL	3351

Figure 90: Interactive Display 1

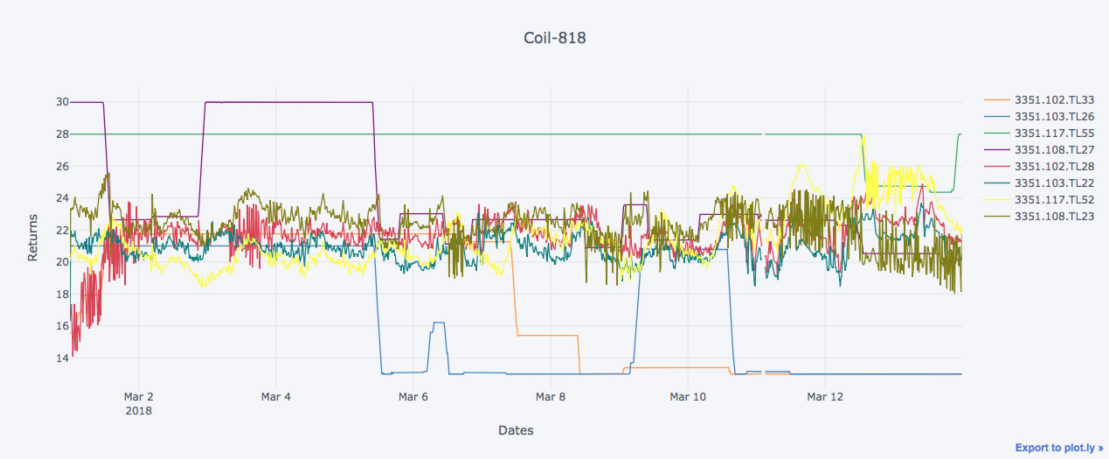

Figure 91: Interactive Display 2

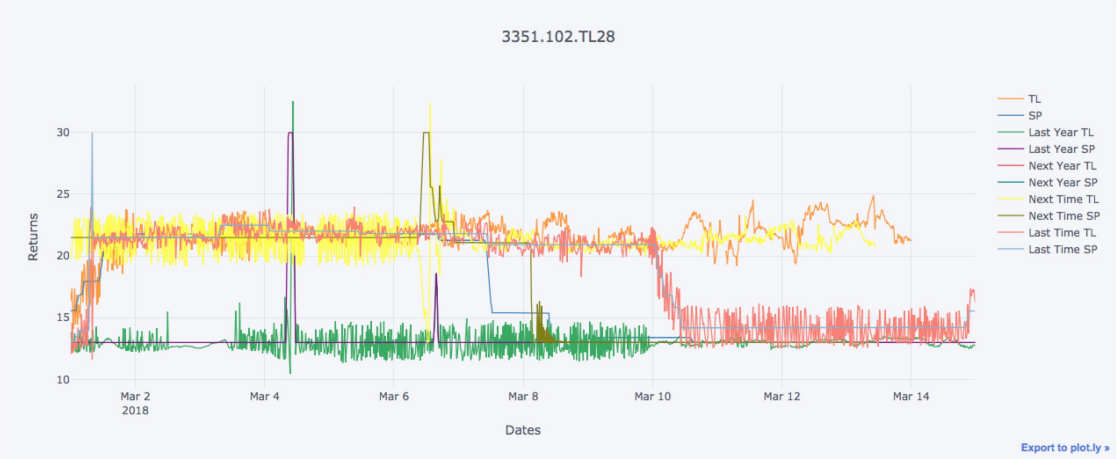

Figure 92: Interactive Display 3

5 Future Work

In this section, potential improvements building on top of the trend log clustering are discussed.

5.1 Automating Anomaly Detection for Room Temperature and Supply Air Temperature Based on Clustering Results

Outliers or anomalous behavior in trend logs can be identified based on their distances from other trend logs. For that reason, clustering can be a useful preprocessing step to anomaly detection as possible approaches explained in the following. As before, we will focus on dynamic changes in room temperature and supply air temperature, but acknowledge that the proposed methods may extend to other types of trend logs.

5.1.1 Interactive display of clustering results for comparison with similar trend logs

A clustering is based on a particular choice of time interval. Now, the idea is to have the user specify an interval where the problem is present and another historical baseline interval, where the problem is supposed to be absent. Based on these periods, two clusterings can be computed and trend logs that change cluster membership are outlier candidates. Interactive display (4.2.3) can then be used to compare trend logs w.r.t. other cluster members to identify candidates as true outliers.

5.1.2 Feature engineering to determine anomaly scores

An anomaly score for each trend log can be based on a) the curve's different trends (e.g. changing slope) within the same period, b) the different changes of a curve relative to other curves closest to it, or c) abnormal (e.g. diverging) trend relative to its set point.

5.2 Simplify the Existing Rules and Redundant Trend Logs

One of the main goals of this report is to work towards simplification of insights generated by Kaizen. To understand the structure of the insights we look at trend logs that are input to the rules that triggered the insights and analyze three aspects of these input trend logs: trend log distribution (93), controller type (94), and keyword from device glossary (95,96,97).

From the first figure below (93), there are 5894 trend logs in the metadata and 69.6% (4100 items) of them are measurement tend logs, the remaining ones represent set points. However, only 22% percent of the measurements are used in rule insights. This could mean that most of the trend logs are irrelevant for fault detection, e.g., due to hierarchical relationships among devices. Also, it could mean that the coverage of trend logs with rules could be improved.

The second figure (94) shows that there are 520 controller IDs in trend log inputs to all of the rule insights. Considering the top 10% of controllers whose trend logs trigger the most rules, these 10% make up almost 35% of all rule insights. Therefore, it is worth exploring whether these 10% of controller rules that generate a lot of insight visibility are in alignment with the controllers' importance and operational state and whether the corresponding controllers or the insight rules need further adjustment.

The last three figures (95,96,97) are using information from a glossary that links trend log names with keywords representing types of controllers, devices, and trend logs. Considering all of the trend logs vs. only the ones that are inputs to rules that have triggered insights, the count of each keyword among the two sets of trend logs are compared with each other. Many of them appear to have an abnormally low proportion of trend logs in rule insights except VAV, HCV, and CCV. This observation might deserve further study.

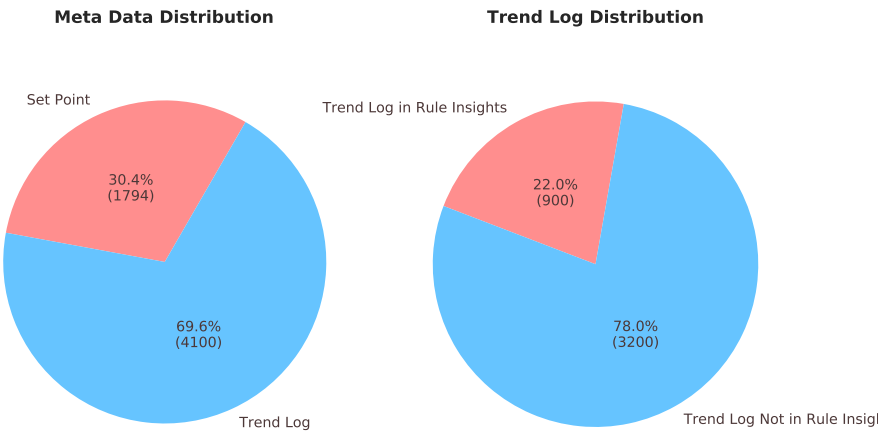

Figure 93: Trend Log Distribution

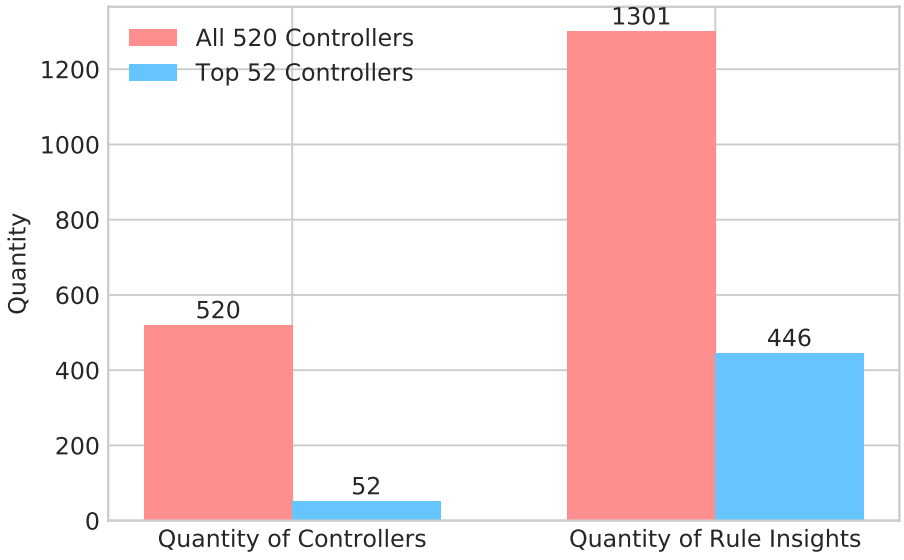

Figure 94: Controllers in Rule insights

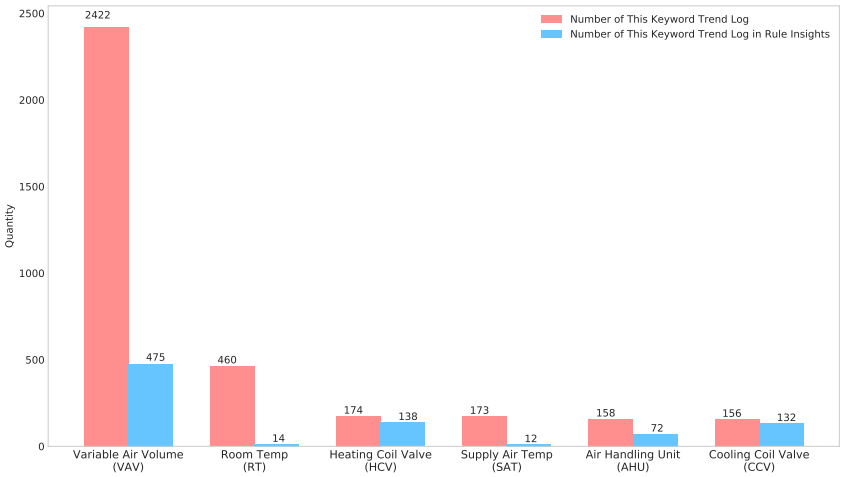

Figure 95: Keywords in Trend Logs VS. in Rule Insights 1

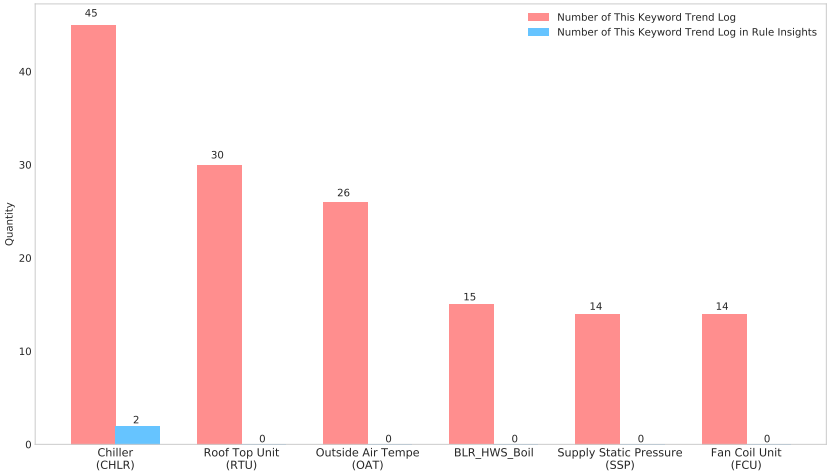

Figure 96: Keywords in Trend Logs VS. in Rule Insights 2

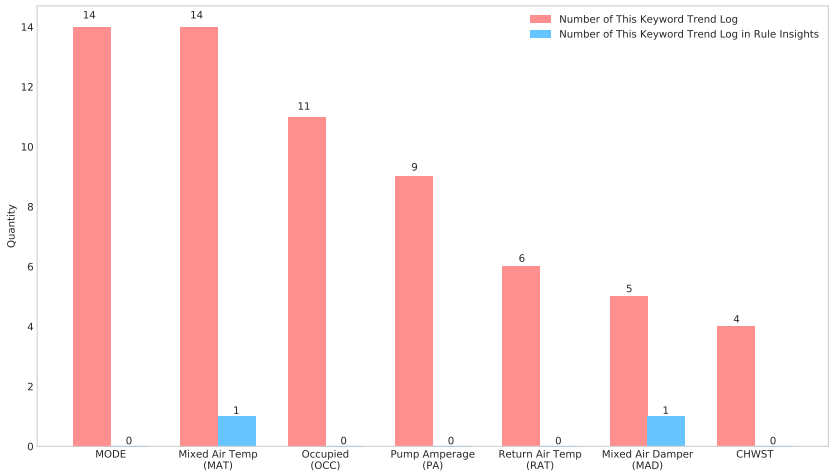

Figure 97: Keywords in Trend Logs VS. in Rule Insights 3