



# Data Analysis Report for UBC Urban Data Lab

2020-06-26

---

Connor Lee, Claudia Nickel, Eva Nguyen & Alex Tamm  
MDS Capstone 2020  
UBCO

## Table of Contents

<b>List of Figures</b>	<b>2</b>
<b>List of Tables</b>	<b>2</b>
<b>Executive Summary</b>	<b>3</b>
<b>Background</b>	<b>4</b>
<b>Motivation</b>	<b>4</b>
<b>Aims &amp; Objectives</b>	<b>5</b>
<b>The Data</b>	<b>5</b>
Data Sources	5
Dataset Used	7
Categories of Data	7
Data Labelling	8
<b>Existing Solutions</b>	<b>9</b>
<b>Project Plan</b>	<b>9</b>
General Approach and Tools Used	9
Details of Approach	10
Details of Each Step	12
Step 1) Cluster NC Data	12
Step 2) Model EC/NC Relationship	13
Step 3) Prep EC Data for Classification Model	14
- Feature Selection	14
- Encode and Scale EC+Metadata	15
Step 4) Classification Model for EC Data	15
Step 5) Output Results and Visualize	16
- Visualization	17
<b>Analysis of Results</b>	<b>17</b>
<b>Difficulties Encountered</b>	<b>18</b>
<b>Conclusion</b>	<b>19</b>
<b>Scope for Future Work</b>	<b>19</b>
Accuracy Updates	19
Performance Updates	20
Scalability Updates	20
<b>References</b>	<b>21</b>
<b>Glossary</b>	<b>23</b>
<b>Appendix A: Detailed Flowchart</b>	<b>24</b>
<b>Appendix B: Clustering Methodology</b>	<b>25</b>
Gower's Distance	25
Multidimensional Scaling	25
Methods of Comparing Clustering Models	25
Clustering Model Selection	26

## List of Figures

<b>Figure 1:</b> Original EWS SkySpark Database	<b>6</b>
<b>Figure 2:</b> UDL SkySpark Database (InfluxDB)	<b>6</b>
<b>Figure 3:</b> Energy Consumption (EC) & Non-Energy Consumption (EC) Sensor Data	<b>8</b>
<b>Figure 4:</b> Project Approach	<b>10</b>
<b>Figure 5:</b> Overview of Project	<b>11</b>
<b>Figure 6:</b> Plots of the Clustering Methods Considered for the Final Model	<b>12</b>
<b>Figure 7:</b> Ridge Regression Coefficients for Each Sensor	<b>14</b>
<b>Figure 8:</b> Grafana Dashboard Screen-Shot	<b>17</b>
<b>Figure 9:</b> Function Flow	<b>24</b>
<b>Figure 10:</b> Examples of Good and Bad Clustering	<b>26</b>
<b>Figure 11:</b> Plots of the Clustering Methods Considered for the Final Model	<b>28</b>

## List of Tables

<b>Table 1:</b> Number of Sensors per End-Use	<b>3</b>
<b>Table 2:</b> Silhouette Scores of Clustering Methods Considered for the Final Model	<b>12</b>
<b>Table 3:</b> Top Performing Clustering Model and Supervised Model Combinations	<b>16</b>
<b>Table 4:</b> Summary of Pharmacy Energy Consumption Sensors Per End-Use	<b>18</b>
<b>Table 5:</b> Clustering Model Comparison	<b>27</b>

## Executive Summary

The UBC Urban Data Lab (UDL) was established to provide open access to sustainability data. UDL provides access to an InfluxDB time series database that contains data on instruments that record the power, energy, water, and gas use of each UBC building. Currently, many instruments have descriptive tags that are not understandable or are too granular for practical use by building managers. For that reason, the focus of the project is to apply machine learning techniques to classify and group instruments by energy end-use. This information will be useful for building managers to easily identify where energy efficiency improvements can be made.

The proposed project was to create a Python program that queries, cleans, and classifies each instrument that meters energy consumption by end-use category. This class of sensor is referred to as an Energy Consumption (EC) sensor in this project. While these EC sensors only make up a small percentage of the total instruments, data from the remaining instruments - referred to as Non-Energy Consumption (NC) sensors - was used to assist with the end-use classification task. The program queried EC and NC data from InfluxDB and the data was fed into a series of models. First, NC data was clustered into groups that reacted similarly. Next, the clustered NC data was used to model the EC/NC relationship. The purpose of modeling the relationship was to develop feature engineering for the EC end-use classification. Afterwards, the model coefficients, instrument metadata, aggregated EC data, and hand-labeled end-use training data were joined together. The final data set was fed into a classification model where EC sensors with unknown end-uses were classified.

As seen in the **Table 1** below, the classification model was able to predict and classify all 208 unknown energy consumption instruments into end-use categories.

**Table 1:** Number of Sensors per End-Use

End-Use Label	Sensor Count	% of Sensors
00_HEATING_SPACE_AND_WATER	54	26%
01_SPACE_COOLING	35	17%
02_HEATING_COOLING_COMBINED	39	19%
03_LIGHTING_NORMAL	26	13%
04_LIGHTING_EMERGENCY	10	5%
05_OTHER	44	21%
Total	208	100%

The project achieved its goal by delivering a Python program that queries, cleans, and classifies instruments by end-use from queried InfluxDB data for the Pharmacy building. The prediction accuracy of the program when applied to the testing dataset was 94.3%. The Python program assists with UDL's vision of assisted Artificial Intelligence (AI) for proactive and preventive maintenance. A few recommendations for future work include increasing the size of the labeled training set and modifying code to work with the updated UDL database.

## Background


The UBC Urban Data Lab (UDL) is located on the UBC Vancouver campus and was established to improve the access, management, and analytics capabilities regarding data generated on the campus. Its primary goals are the following [1]:

1. "Provide open access of UBC sustainability data to researchers, policymakers and operational staff;"
2. "Support the monitoring and measurement of sustainability performance for buildings, transportation, biodiversity specifically as it relates to the policy commitments of UBC Sustainability Initiative (USI) and Campus and Community Planning (C&CP)."

Urban Data Lab has access to a platform managed by UBC Energy and Water Services (EWS) that stores the power, energy, water, and gas usage of each UBC campus building. UDL is mirroring a portion of this information on a publicly accessible database referred to in this document as the UDL InfluxDB SKYSPARK database. The data stored on UDL's database aids in their research and operational interests of energy benchmarking, building retrofits, end-use analysis, climate scenarios, equipment maintenance, and occupancy analysis. This capstone project was concerned specifically with end-use analysis as it relates to UDL's database.

## Motivation

This project was motivated by UDL's goal for students, researchers, and operational staff to better understand the different types and quantities of energy end-uses in a building - such as lighting, HVAC, and electrical plugs. While data on energy use in buildings was already available through UDL's InfluxDB SKYSPARK database, the end-use categories of this energy generally were not available or not clearly identified. To achieve their goal of helping people understand energy end-uses, UDL would like to apply machine learning techniques for classification, energy profiling, and energy forecasting by end-use [1]. Understanding which instruments are associated with each end-use will allow UDL and other researchers to analyse the campus wide environmental impact of upgrading specific equipment and systems. In addition, this insight will assist decision makers on campus with improving energy efficiency. UDL requested a data science capstone project on this topic to help with some of the initial analysis and groundwork. The primary issue was that UDL was unsure of which instruments were for which end-use. The reason for this is most of the instruments were installed before any documentation was created. The lack of documentation caused uncertainty of which instrument belonged to which end-use. Another issue was that instruments in specific buildings (i.e. the Pharmacy Building) had tags that were not understandable or were too detailed for practical use by building managers and other interested parties. For example, it may have been clear what specific piece of equipment



the sensor was associated with, but the equipment's purpose was unclear and the descriptive information may not have been formatted in a general way beyond specific equipment id code. UDL's immediate interest was in having relevant sensors be given an end-use category. By classifying sensors by end-use, it helps group the unclear/overly-detailed sensors into a similar framework for analysis. Ultimately, this information will be useful for building managers who need to be able to quickly and easily identify which systems in a building use the most energy and where improvements can be made in terms of energy efficiency. Furthermore, this work will be considered a small step towards UDL and UBC's grander vision of assisted artificial intelligence (AI) for proactive and preventative maintenance.

## Aims & Objectives

The question which guides this project is,

*"Based on a building's sensor data, how can the data be grouped  
automatically into end-use classification?"*

No existing approaches to solving this question were identified during initial literature review, which necessitated testing different models and methods in order to achieve an optimal result.

The primary objective was to create a program that is able to query building sensor data from UDL's InfluxDB SKYSPARK database, apply a series of cleaning and feature engineering functions to prepare the data for classification, and then classify each sensor that measures energy usage as the specific type of end-use (i.e. lighting, heating, electrical outlets, etc). Additionally, these energy usage values, grouped together by their newly applied end-use categories, were to be visualized in a Grafana web-based dashboard.

## The Data

This section discusses the dataset, how the data was broken down and how the data was labelled for training and testing sets.

### Data Sources

UDL provides public access to two databases through InfluxDB, one is the ION database and the other is the SKYSPARK database. The SKYSPARK database mirrors the UBC EWS SkySpark database; it provides data that was recorded by smart devices and meters in each of the UBC buildings [2]. This database is updated every 15 minutes with smart device and meter data, and every hour with weather data [3]. The database elements are named in conformance with the Project Haystack tag guidelines [4]. UDL provides public access to the mirrored ION and SkySpark databases in order to provide researchers and students access



to UBC building data, without overloading the UBC EWS databases with queries. **Figure 1** shows a small subsection of the data coming from the original EWS SkySpark database, with only a few of the columns shown, and **Figure 2** shows the data from UDL's InfluxDB SKYSPARK database, with all columns shown. As you can see, the data coming from the EWS SkySpark database in **Figure 1**, includes a large amount of metadata, most of which is not fully displayed below. Most of the columns are boolean to indicate if that sensor is On/Off or if it is a part of something (i.e. if that sensor belongs to an air handling unit or not). The data in **Figure 2** is primarily what was used and it contains 8 columns that includes data on each sensor reading. Each sensor reading has a unit of measurement, a timestamp for when the reading was recorded, a location of the sensor, such as which building, floor and room it is located in and information on which piece of equipment it belongs to (i.e. HVAC).

id	ahu	ahuMode	air	alarm	avg	bacnetConnRef	bacnetCur	bacnetHis	bacnetObjectid	bac
① Pharmacy Heating Plant HX2 P-HX2A HX2_PHX2A_VFD_PWR(kW)	○					PHARMA PHARM_HX2_FCU_B05 (701100)		TL56		✓
① Pharmacy Elec Submeters LEED-2N1PC3 2N1PC3_CurrentC						PHARMA PHARM_LEED_METER_MOD2 (702100)		TL59		✓
① Pharmacy Rm Corr FC-513 FCU_513_S			✓			PHARMA PHARM_FCU_513 (700742)		TL1		✓
① Pharmacy Rm 1420 FC-111 FCU_111_S			✓			PHARMA PHARM_FCU_111/FF_102/EF_106 (701206)		TL1		✓
① Pharmacy Elec Submeters LEED-6ETLE1 6ETLE1_CurrentB						PHARMA PHARM_LEED_METER_MOD1 (702000)		TL208		✓
① Pharmacy Rm B503 EAV-B5048 B5048_AVG_SPACE_TEMP_AV			✓			PHARMA PHARM_MACRO_SERVER_FH_TEMP_RM (702900)		TL182		✓
① Pharmacy AHU-15 AHU15_FIRE_MODE						PHARMA PHARM_AHU14_15_EF3_SB1~2 (700900)		TL122		✓
① Pharmacy Rm 6107 FC-601 FCU_601_SCHED						PHARMA PHARM_FCU_601 (700514)		TL9		✓
① Pharmacy Unsorted Points CO2_RM_B212						PHARMA PHARM_MISC_BSMNT_AL (700758)		TL3		✓
① Pharmacy CRAH System CRAH-2 CRAH2_LO_RT_AL_BV				✓		PHARMA PHARM_HTRE_JHWS_HX3_CRAH (700200)		TL178		✓
① Pharmacy Rm 4616 RAD-4-09 RZ4_09_HWRT						PHARMA PHARM_RZ4_04&09 (701340)		TL11		✓
① Pharmacy Heating Plant BLR-3 BLR_PB3_S						PHARMA PHARM_BLR1~4_CT_HX1_7_DHW (700800)		TL47		✓
① Pharmacy EAV-BE001 BE001_SASH_OPEN_PERCENT						PHARMA PHARM_MACRO_SERVER_FH_TEMP_RM (702900)		TL360		✓
① Pharmacy LEF-2 EF-2C EF2_F3_VFD_INST_PWR(kW)						PHARMA PHARM_SB3~6_EF2_SMOKE_EF14~16 (701600)		TL144		✓

**Figure 1:** Original EWS SkySpark Database

	time	equipRef	groupRef	navName	siteRef	typeRef	unit	value
0	2019-12-13T22:24:37Z	AHU-02	CIRS Air Systems	Discharge Air Temp	CIRS	CIRS_AHU2_SUPPLY_AIR_T	°C	16.707474
1	2019-12-13T22:39:37Z	AHU-02	CIRS Air Systems	Discharge Air Temp	CIRS	CIRS_AHU2_SUPPLY_AIR_T	°C	16.105682
...	...	...	...	...	...	...	...	...
580	2019-12-19T23:39:59Z	AHU-02	CIRS Air Systems	Discharge Air Temp	CIRS	CIRS_AHU2_SUPPLY_AIR_T	°C	23.058758
581	2019-12-19T23:54:59Z	AHU-02	CIRS Air Systems	Discharge Air Temp	CIRS	CIRS_AHU2_SUPPLY_AIR_T	°C	23.049675

**Figure 2:** UDL InfluxDB SKYSPARK Database

Unfortunately, the value field in UDL's InfluxDB database was initially stored as a string data type as opposed to numeric values being stored as a float, boolean values as a boolean, strings as a string, etc. This structural issue is discussed in more detail in the **Difficulties Encountered** section of this report but the major impact was that all data over the date range of interest needed to be queried, aggregated, and formatted locally instead of using InfluxDB's standard aggregation functions to reduce the amount of data queried. Due to this large amount of data transferred, results had to be stored in individual text files to be later ingested into the program via a series of functions that aggregated and formatted the data for modeling use.

## Dataset Used

The entire UDL InfluxDB database's UBC\_EWS measurement at time of writing consisted of 499,521,763 individual records stored in 54,692 distinct series. These records were split across 128 buildings and covered a range of times with the earliest sensor reading dating back to 2017-05-27. To create a more manageable subset of the data, UDL recommended that the Capstone Team focus on the Pharmacy building as it was a modern building with quite a few sensors and the majority of the descriptive tags for the sensors had been checked and cleaned by EWS. The date range chosen was 2020-01-08 through 2020-06-08. While data was available for the Pharmacy building before January 8th, there were changes in SkySpark on January 7th that impacted a significant number of the Pharmacy building sensors and caused the UDL InfluxDB database to treat them as a separate set of sensors (more on this in the **Difficulties Encountered** section). The chosen start date avoided causing multiple instances of every sensor in the dataset. The end date was simply the cut-off day for when the dataset needed to be finalized for testing the models. Due to server outages, there are several days with no readings and other days with fewer than normal readings, however the dataset encompasses a 153-day span of data for the Pharmacy building and when stored in comma-separated values format it is approximately 9GB in size.

## Categories of Data

The sensor data used in the project was split into two categories of sensors: Energy Consumption (EC) sensors and Non-Energy Consumption (NC) sensors. EC sensors are sensors and meters that record an energy use reading (such as kilowatts per hour consumed by a piece of equipment). NC sensors are the more broad category and include every type of sensor that is recording something other than energy use, such as temperature, fan speeds, damper opening percentage, occupancy sensors, etc. **Figure 3** below shows EC sensors in red and NC sensors green.



	equipRef	groupRef	navName	siteRef	typeRef	unit	value	
2020-05-31 06:53:17-07:00	Rm 5202 EAV-5E068	Pharmacy Floor 5	Exhaust Air Flow High Lim Sp	Pharmacy	5E068_VLV_FLOW_FDBK_HILIM_SP	L/s	450.000000	NC Data
2020-05-31 07:08:18-07:00	Rm 6311 EAV-6E049	Pharmacy Floor 6	Exhaust Air Flow High Lim Sp	Pharmacy	6E049_VLV_FLOW_FDBK_HILIM_SP	L/s	250.000000	
2020-05-31 09:29:37-07:00	Rm 3335 VAV-3S035	Pharmacy Floor 3	Zone Temp Effective Sp	Pharmacy	VAV_3S035_RT_SP	°C	23.000000	
2020-05-31 01:30:00-07:00	Rm 4130 FC-403	Pharmacy Floor 4	Zone Temp Effective Sp	Pharmacy	FCU_403_RT_SP	°C	21.500000	
2020-05-31 09:01:23-07:00	Rm 3202 VAV-3S015	Pharmacy Floor 3	Discharge Air Damper Open Cmd	Pharmacy	VAV_3S015_Dmp_Open	omit	True	
2020-05-31 10:59:42-07:00	Windows	Pharmacy Floor 5	L5_SE_OAT_CLG_REQUEST	Pharmacy	L5_SE_OAT_CLG_REQUEST	omit	True	
2020-05-31 09:38:23-07:00	Elec Submeters LEED-6N4LW1	Pharmacy Utilities	6N4LW1_EnergyPosSum	Pharmacy	6N4LW1_EnergyPosSum	kWh	59165.832031	EC Data
2020-05-31 09:45:00-07:00	AHU-01 SF	Pharmacy Air Systems	Energy	Pharmacy	AHU1_SF_VFD_PWR(kWh)	kWh	10840.208008	
2020-05-31 04:45:00-07:00	Cooling Plant P-9A	Pharmacy Hydronic Systems	Energy	Pharmacy	CHWP_P9A_VFD_PWR(kWh)_TL	kWh	3164.388672	
2020-05-31 00:39:54-07:00	Elec Submeters LEED-ATS-S3	Pharmacy Utilities	ATS-S3_EnergyPosSumNR	Pharmacy	ATS-S3_EnergyPosSumNR	kWh	20206020.000000	
2020-05-31 04:58:19-07:00	Elec Submeters LEED-ATS-DCB	Pharmacy Utilities	ATS-DCB_EnergyPosSum	Pharmacy	ATS-DCB_EnergyPosSum	kWh	4129881.250000	
2020-05-31 04:15:00-07:00	LEF-3 EF-3A	Pharmacy Air Systems	Energy	Pharmacy	EF3_F1_VFD_PWR(kWh)	kWh	43112.464844	
2020-05-31 04:28:32-07:00	Elec Submeters LEED-CH-2	Pharmacy Utilities	CH-2_EnergyPosSum	Pharmacy	CH-2_EnergyPosSum	kWh	2470925.500000	
2020-05-31 09:30:00-07:00	Cooling Plant P-9B	Pharmacy Hydronic Systems	Energy	Pharmacy	CHWP_P9B_VFD_PWR(kWh)_TL	kWh	3147.771729	

**Figure 3: Energy Consumption (EC) & Non-Energy Consumption (NC) Sensor Data**

While it may not be obvious from the sample shown in **Figure 3**, EC sensors are only a very small subset of our dataset (and the database as a whole). The project's dataset contained 208 EC sensors and roughly 8000 NC sensors, with the breakdown assumed to be similar for other UBC buildings.

## Data Labelling

To create data for training and testing different classification model candidates, as many EC sensors as possible were labelled by hand with end-use categories. The original goal was for end-use labels to follow Natural Resource Canada's (NRCan) secondary energy end-use categories, however it was not possible to identify EC sensors for each NRCan category. An alternative list of energy end-use categories was created to represent all the groupings present in the dataset. The labelled data was later split into a training and test set using a 80:20 split. Energy consuming sensors were hand labelled with the following end-use categories:

- 00\_HEATING\_SPACE\_AND\_WATER
- 01\_SPACE\_COOLING
- 02\_HEATING\_COOLING\_COMBINED
- 03\_LIGHTING\_NORMAL
- 04\_LIGHTING\_EMERGENCY
- 05\_OTHER
- 99\_UNKNOWN

## Existing Solutions

Initial investigations into existing solutions for this particular problem did not reveal anything of note, however, investigation into related topics did provide some useful insight into how to identify energy consumption sensor end-uses. The most relevant existing solutions found were related to energy benchmarking, and clustering functional data.

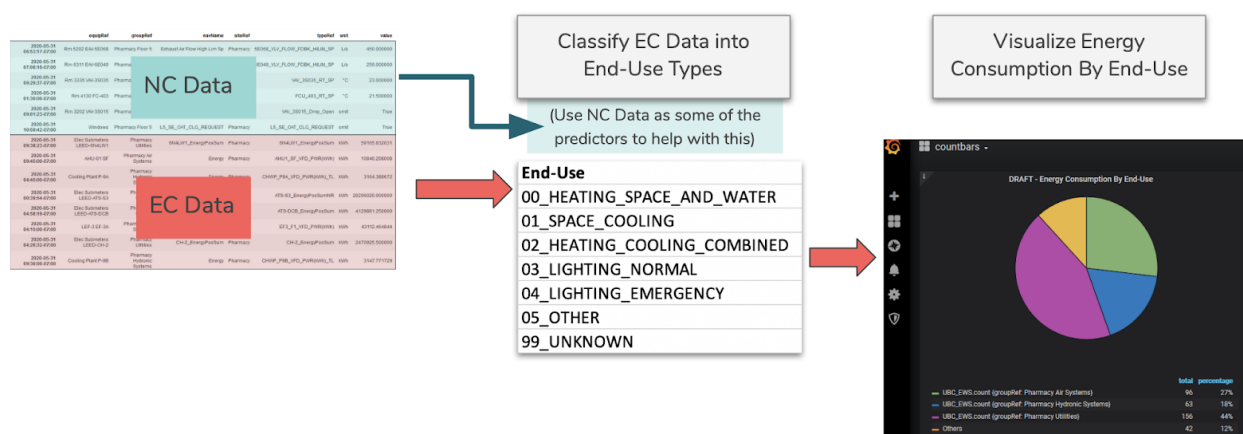
Improving energy efficiency on campus requires a thorough understanding of the energy consumption of each building on campus. Energy benchmarking is used for analyzing energy performance for buildings of similar types [5]. While the goal of this project is to identify sensor end-uses based on similar performance and not to benchmark the buildings, the similarities in the type of data being analyzed in both situations suggests that similar methods of analysis could apply. There are several benchmarking methods, of which many (such as points based systems) would not be well suited for analysis and classification of sensors. The application of clustering for energy benchmarking as done in [5] was the most relevant method of energy benchmarking found for application to sensors rather than buildings. Clustering is primarily used to identify similarities between observations, in the case of this project it can be used to identify sensors that have similar types of responses.

The available data for this project was time series data for a large number of sensors within buildings. Since the primary focus of this project was to identify the end-uses of a subset of the sensors, it was more beneficial to think of the data in a functional form, where the sensors whose end-uses were not being predicted (the NC sensors) were used as predictors for the EC sensors, rather than as a set of independent time series. Applying the underlying concept of clustering functional data on estimated regression coefficients as discussed in [6] allows for the identification groupings of sensors that behave similarly. While Tarpey's article focuses on using the k-means clustering model, Tarpey concludes the paper with a discussion on the shortfalls of the k-means clustering method [6]. Tarpey goes on to recommend implementing other clustering models such as finite mixture models in order to avoid the issues inherent to the k-means model [6].

## Project Plan

### General Approach and Tools Used

The general approach that was adopted for solving UDL's question of how to classify energy sensor end-uses in their database is shown below in **Figure 4**.



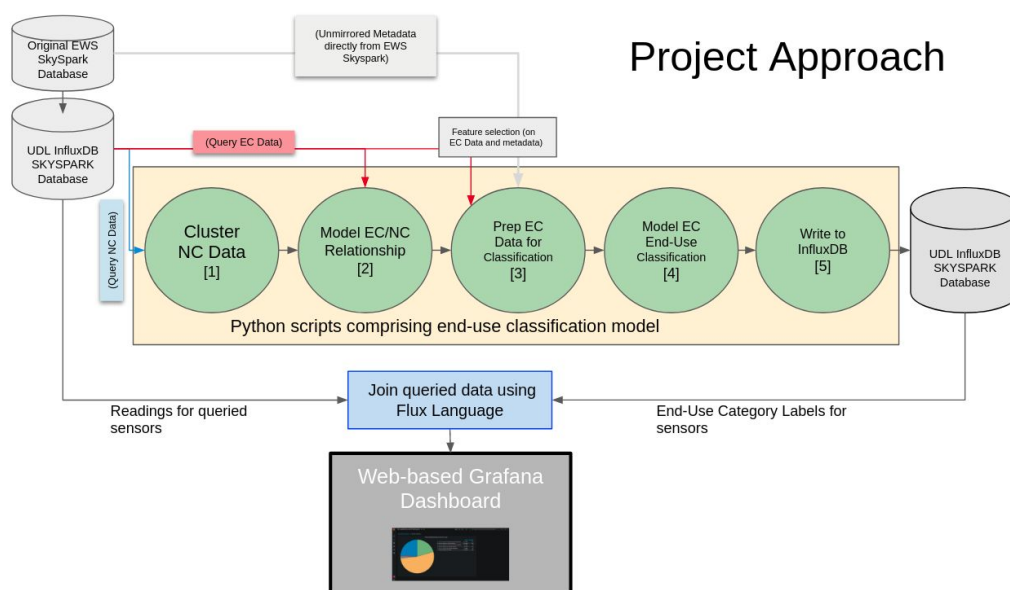
**Figure 4:** Overview of Proposed Solution

In this simplified overview of the proposed solution, the NC and EC sensor data queried from UDL's InfluxDB SKYSPARK database is shown flowing into an abstraction of the program that has been written to work with the data and model end-uses for the EC sensors before feeding the results to a visualization dashboard.

The tools employed to implement this solution include the Python programming language (version 3.7x) with the common data analysis packages; Pandas, Numpy, and SciKit-Learn, and the influxDB-python package for connecting to UDL's database. The visualization dashboard was created using Grafana open-source software deployed on a free-tier of the commercial Grafana Cloud service.

### Details of Approach

To implement the general solution discussed above, multiple models and concepts had to be tested; however, the flowchart shown in **Figure 5** lays out the general framework of the approach and how data flows between different components.



**Figure 5:** Framework of project

This flowchart encompasses the full scope of the program including; data sources on the left, the data preparation and modeling code at the heart of the end-use classification solution in the yellow box, and finally the various outputs required for visualization of the results in a dashboard. The general concept was to pass the queried data from the databases into models and functions in steps 1 through 3 to create and/or select features that characterize the EC sensors. Using these relevant features, step 4 of the program then assigns end-use categories to EC sensors based on if they matched the characteristics of EC sensors that already had known end-use categories. Step 5 was to store the resulting EC sensor end-use labels and access them with the visualization dashboard.

Due to data sources being injected into the process at different steps (see **Figure 5**), varying amounts of data cleaning and manipulation were necessary within each step of the program, thus that particular topic is discussed as appropriate in the relevant sections.

## Details of Each Step

Information on implementation details and choices behind the techniques used in the final product are broken down in the following steps. If the reader would like additional detail on what is covered below, please see **Appendix A** for a more detailed flow chart of the process.

### Step 1) Cluster NC Data

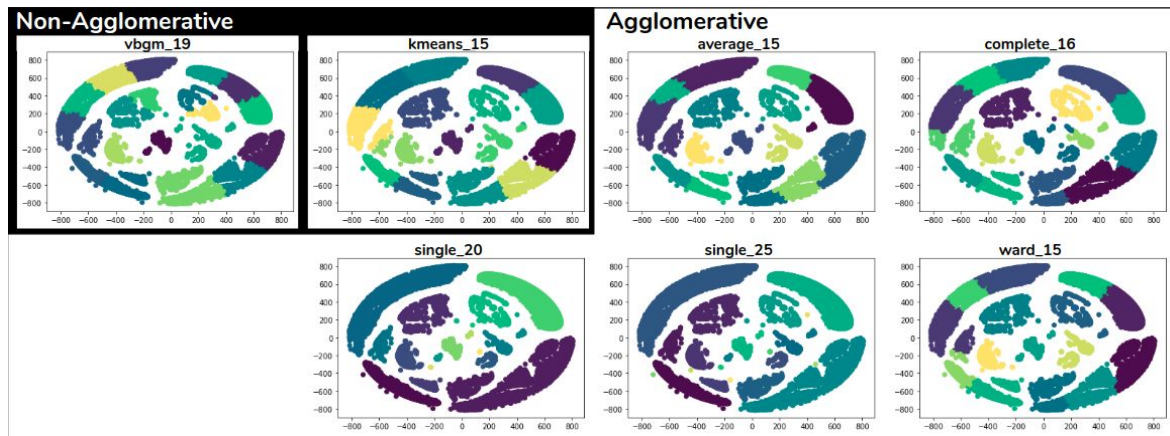
Clustering is the process of identifying groups of observations that are more similar to the other observations within their cluster than they are to other observations outside of their cluster. The need for clustering arose from the fact that there were thousands of NC sensors in each building, therefore it was computationally infeasible to relate each NC

sensor to each EC sensor. By clustering the NC sensors into groups of sensors that react similarly, it was possible to identify how the EC sensors change with respect to a group of NC sensors. For example, if a given cluster represents occupancy sensors, the measurements those sensors are providing are more likely to be related to lighting than water heating. As a result, the clustering of similar sensor responses was a form of feature reduction, that allowed for the measurements of each NC sensor to be accounted for, without making it computationally infeasible to train the model. Gower's distance and multidimensional scaling was used on the data in order for it to be fed into the models. More details on how Gower's distance and multidimensional scaling were used is located in **Appendix B**. Explanations of how the varying cluster models were compared and the details on the actual model selection are also outlined in **Appendix B**. Silhouette scores and observing the clarity of clusters when plotted were used as the main means for comparison between models. The Silhouette Scores for each of the clustering models that were selected are shown in **Table 2** below.


**Table 2:** Silhouette Scores of Clustering Methods Considered for the Final Model

Model	Number of Clusters	Silhouette Score
Agglomerative with Average Linkage (average_15)	15	0.465
Agglomerative with Complete Linkage (complete_16)	16	0.454
Agglomerative with Single Linkage (single_20)	20	0.266
Agglomerative with Single Linkage (single_25)	25	0.178
Agglomerative with Ward Linkage (ward_15_)	15	0.476
K-Means (kmeans_15)	15	0.501
Variational Bayesian Estimation of a Gaussian Mixture (vbgm_19)	19	0.433

Plots of the clusters resulting from each of the models listed in **Table 2** are shown in **Figure 6** below.



**Figure 6:** Plots of the Clustering Methods Considered for the Final Model



From **Table 2** above it is clear that the K-Means model has the best silhouette score, suggesting that it provides the most clearly defined clusters; however, from observation of the plots shown in **Figure 6** it appears as though the Agglomerative clustering model with Single Linkage and 25 clusters provides the best separation of clusters despite having the lowest silhouette score. This inconsistency is discussed further in **Appendix B**. The clustering model selected for implementation into the final model was Agglomerative clustering with Single Linkage and 20 clusters. This selection was made in conjunction with the selection of the supervised model for implementation into the final model and will be discussed further in the **Classification Model for EC Data** section below.

### Step 2) Model EC/NC Relationship

The purpose behind modeling a relationship between the EC and NC sensors was to use the NC sensor data (clustered in the previous step to provide a more manageable dataset) to create a set of features that could help characterize a given EC sensor. EC sensors with similar characteristics could then be classified with similar end-use categories in step 4 of the program. As there were relatively few EC sensors in the dataset and very little information to characterize them, this feature engineering step was critical to providing enough information for the classification model to be effective. To clarify how this relationship can provide valid characteristics of an EC sensor, one can consider a cold winter day: the cluster of NC sensors including exterior temperature would show low values, the cluster of NC sensors including heater system activity (i.e. gas boiler fire rate) would show increased activity, and the EC sensors measuring energy used for heating would read a much higher energy use rate than on a hot summer day. In the previous example, it would be expected that all of the heating-related EC sensors would have a relatively similar EC/NC relationship, compared to, for example, EC sensors for lighting that have no link to exterior temperature and heating system activity.

The relationships between all the NC sensor clusters' readings and each EC sensor's readings were modeled using ridge regression. The resulting coefficients for each model, that is the best fit of the NC data cluster's values (predictor variables) to the given EC sensor's values (response variable), were used to represent the relationship between these two groups of sensors and were included with the features fed into the classification model.

The lasso regression method was originally considered; however, due to issues with achieving convergence with the given data, ridge regression was used instead. The key difference between lasso and ridge regression is the penalty term. Ridge regression's penalty term is the sum of squares of coefficients whereas lasso's penalty term is the sum of absolute values of coefficients.

As mentioned previously, data from the Cluster NC Data step and data from EC sensors were used as inputs to this model. Clustered NC Data results include aggregated values



(mean, standard deviation, min, max, update rate) for each cluster group by date and hour. Similarly, EC sensors data included aggregated mean values per unique sensor (uniqueID) by date and hour. The two data sets were joined on date and hour. For each unique EC sensor id in the merged data set, a ridge regression model was created. The response variable was the EC mean value and the predictor variables were the NC clusters. The resulting coefficients of the fitted ridge regression model for each unique sensor were combined and stored in a table as shown below in **Figure 7**.

**Equation Generated from Ridge Regression for each sensor**

$$ec\_sensor\_response = \beta_0 + \beta_1 mean_{c1} + \beta_2 max_{c1} + \beta_3 min_{c1} + \beta_4 std_{c1} + \beta_5 u\_rate_{c1} + \dots + \beta_{5+n-4} mean_{cn} + \beta_{5+n-3} max_{cn} + \beta_{5+n-2} min_{cn} + \beta_{5+n-1} std_{cn} + \beta_{5+n} u\_rate_{cn}$$

0	1	2	3	...	5n-2	5n-1	5n	uniqueID
0.000037	-0.004377	0.0	-0.000041	...	5.876493	8.502804	20.087383	AHU-01 SF Air Systems Energy AHU1_SF_VFD_PWR(kWh)
0.000039	-0.004622	0.0	-0.000044	...	6.537176	8.851925	20.473544	AHU-02 SF Air Systems Energy AHU2_SF_VFD_PWR(kWh)

**Figure 7:** Ridge Regression Coefficients for Each Sensor

### Step 3) Prep EC Data for Classification Model

This step prepared EC data to feed into the classification model by appending various features, passing it through a feature selection process, encoding the categorical features, scaling the numeric features, and combining the columns of coefficients from the EC/NC Relationship model results. First, it required taking EC data that had been aggregated by unique sensor id and combining it with the metadata. The prepped EC data was created by joining the two data sets on unique sensor id. Subsequently, fields that were not useful for classification were dropped and feature selection techniques were applied to the resulting data. After feature selection techniques were applied and unwanted fields were dropped, categorical fields were encoded and numeric fields were scaled. The last step was to join this data set with the ridge regression coefficients on unique sensor id. After which the final data set was ready to be fed into the classification model. Data cleaning for this step included: changing the boolean check marks in the metadata into more descriptive categorical levels, fixing any incorrect units of measurements, and dropping duplicate sensor ids in metadata.

#### Feature Selection

Feature selection for categorical fields used the Mutual Information method and feature selection for continuous fields used the ANOVA method. Cross-validation was used to identify the most optimal value for k in both methods.

### Encode and Scale EC+Metadata

Categorical fields were encoded by designating a value of 1 to sensors that had the categorical observation and a value of 0 otherwise. Encoding was required because machine learning models require numeric input. The continuous fields were encoded by using a MinMaxScaler in order to scale each observation between the range of 0 and 1. Scaling assisted in improving model accuracy.

### Step 4) Classification Model for EC Data

The classification model for the EC sensors is the portion of the modeling phase where the actual energy end-use classification occurred. This portion of the model used the features in the prepared data from the previous steps to train a supervised learning model that predicted the EC sensor end-use category. The training set was developed by hand labeling all of the EC sensors from the Pharmacy building into the categories listed in the **Data Labelling** section above. Supervised learning was selected as the preferred classification method since, after hand labeling the data, a training set with specific end-use categories was available. Supervised learning models lend themselves well to classifying datasets into a predetermined set of classes. Whereas unsupervised methods focus more on identifying groups of observations with similar behaviours and require additional analysis and interpretation of the clusters in order to identify what class the cluster belongs to.

Each of the following supervised models were trained and tested on data from each of the clustering methods discussed in **Appendix A**:

- AdaBoost
- Bagging
- Decision Tree
- Extremely Random Trees
- Gradient Boost
- K-Nearest Neighbours (KNN)
- Kernel SVM
- Logistic Regression
- Naive Bayse
- Random Forest

The top performing supervised model for each clustering method as defined by the accuracy, precision, recall, F1 score and log loss model comparison metrics are highlighted in green in **Table 3** below.

**Table 3:** Top Performing Clustering Model and Supervised Model Combinations

Clustering Model	Linkage Method	Number of Clusters	Supervised Model	accuracy	precision	recall	f1_score	log loss
Agglomerative	Average	15	Bagging	0.9143	0.9310	0.9143	0.9144	0.2683
Agglomerative	Complete	16	Random Forest	0.9429	0.9524	0.9429	0.9449	0.3786
K-Means	-	15	Extremely Random Trees	0.9143	0.9310	0.9143	0.9144	1.2282
Agglomerative	Single	20	Bagging	0.9429	0.9524	0.9429	0.9449	0.1961
Agglomerative	Single	25	Gradient Boost	0.9429	0.9524	0.9429	0.9449	0.1979
VBGM	-	19	Gradient Boost	0.9429	0.9490	0.9429	0.9389	0.4589
Agglomerative	Ward	15	Bagging	0.9143	0.9310	0.9143	0.9144	0.3518

From **Table 3** above it is clear that the combination of Agglomerative Clustering with Single Linkage and 20 clusters for the clustering model, and Bagging for the supervised model was the optimal model for the given data as it provided the highest accuracy, precision, recall and F1 score and the lowest log loss.

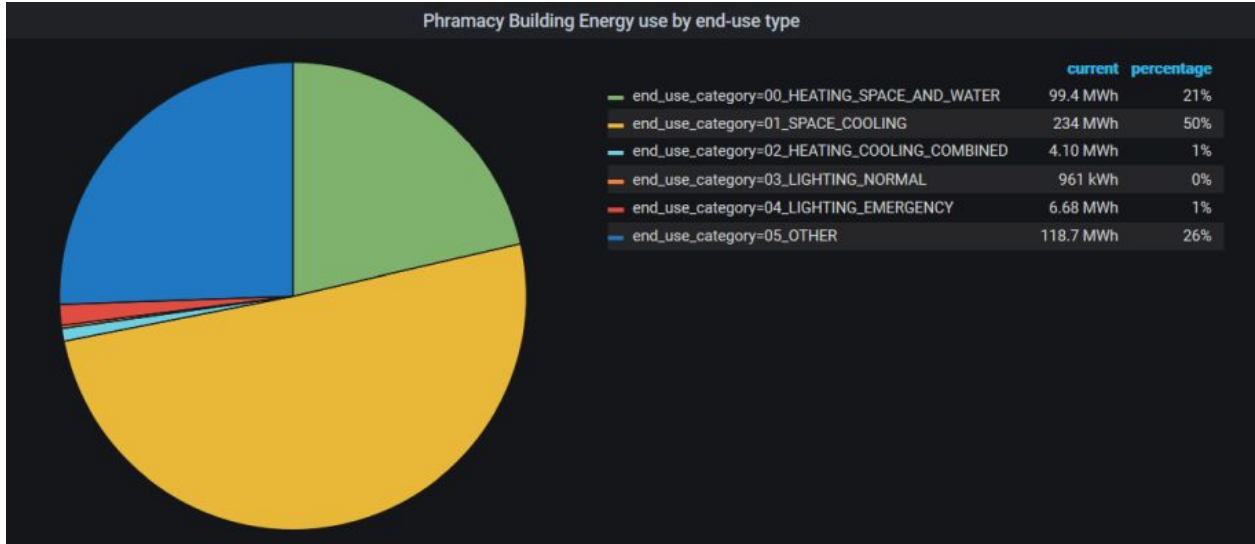
### Step 5) Output Results and Visualize

The requested Grafana dashboard required access to energy consumption readings by sensor and end-use category labels for those sensors. The EC sensor readings were already accessible in UDL's InfluxDB SKYSPARK database and at UDL's request, the end-use labels were stored in a separate `END\_USE` measurement (similar to a "table" in relational databases) in the same database. While the end-use label data was not well-suited to a time-series database, the time and effort of setting up and hosting a separate relational database on UDL's server was not justified for this project.

Points (records) were written to the UDL InfluxDB SKYSPARK database using Python and the same influxdb-python package used to query data for the main modeling tasks. The code to do this process resides in a separate file from the main code named write\_enduse\_to\_influx.py. This simple program reads the output of the main code file (i.e. the output from Step 4) from a comma-separated values file of unique sensor IDs and their associated end-use labels. No data cleaning was necessary at this point in the process but a timestamp with a constant value of '2020-01-01' needed to be attached to the data before the program could write data to the database. Due to using a constant value for the timestamp, any future updates to the InfluxDB database with the same uniqueID values will overwrite existing end-use label data for that uniqueID with the new value. This step was split into a separate file to make initiating the irrevocable write/update process a more intentional action on the part of the user. Note that while UDL provided our team write-access to the InfluxDB database, the login credentials must be inputted by the user when running the program and are not stored in the code.

## Visualization

Grafana open-source software was used to create a web-based dashboard with a pie-chart of energy consumption by end-use category, screen-shot shown below in **Figure 8**.



**Figure 8:** Grafana Dashboard Screen-Shot

As mentioned in the previous section, this proof-of-concept visualization pulls energy consumption readings from the UDL InfluxDB SKYSPARK database - specifically from the older `UBC\_EWS` measurement. The query also pulls data from the `END\_USE` measurement in the same database, joins the data by sensor id (`uniqueID` tag) and groups by end-use category (`endUseLabel` field). The Flux query language was used to do this, using Grafana Lab's InfluxDB (Flux) Datasource plugin to communicate with InfluxDB.

The Grafana dashboard has been hosted on UDL's Grafana Cloud account with an exported .json file of the dashboard and extensive documentation markdown file included in the project's Git repository.

## Analysis of Results

The selected clustering and supervised model combination of Agglomerative clustering with Single Linkage and 20 clusters for the clustering model and Boosting for the supervised model provided a 94.3% accuracy when applied to the testing dataset. This prediction accuracy, while not perfect was achievable in significantly less time and with significantly less effort than manually labeling the data. Furthermore, as discussed in the **Motivation** section, many of the EC sensors cannot be manually labeled as they are not labeled in a human-readable format.

Upon running the model on the full set of EC sensors within the Pharmacy building, each EC sensor was provided an end-use category. The number of EC sensors per end-use category within the Pharmacy building are shown in **Table 4** below.

**Table 4:** Summary of Pharmacy Energy Consumption Sensors Per End-Use

End-Use Category	Sensor Count	% of Sensors
00_HEATING_SPACE_AND_WATER	54	26%
01_SPACE_COOLING	35	17%
02_HEATING_COOLING_COMBINED	39	19%
03_LIGHTING_NORMAL	26	13%
04_LIGHTING_EMERGENCY	10	5%
05_OTHER	44	21%
<b>Total</b>	<b>208</b>	<b>100%</b>

The largest grouping of EC sensors, consisting of 26% of the total EC sensors within the Pharmacy building, was for space and water heating, while the smallest group of sensors, consisting of 5% of the sensors within the Pharmacy building, was for emergency lighting. This makes sense as heating equipment typically has larger power draw than lighting, as a result they are typically measured at a more granular level. Whereas lighting power consumption is typically measured at the level of a single lighting electrical panel, which typically feeds a large number of individual lights. Furthermore, emergency lighting is typically much more sparse within buildings than normal lighting as it requires minimal power draw while still providing sufficient lighting for people to exit the building during an emergency. In contrast, normal lighting is intended to fully light up rooms and corridors.

## Difficulties Encountered

Throughout the project there were several difficulties and roadblocks encountered.

- Initial misunderstanding that all sensors needed to be assigned an end-use label. When we were informed that only a small subset of the sensors needed to be assigned an end-use label, it necessitated a redesign of the planned model .
- Once we had a clear idea of which sensors did need to be hand labeled, it took a lot of time to understand the data well enough to label it.
- InfluxDB and its influxQL language is designed to store and work with time-series information. Trying to join additional tags of information necessitated learning and enabling the newer query language that InfluxDB is transitioning to in the major release of the product (Flux).
- The UDL InfluxDB SKYSPARK database had been storing all sensor readings as a string data type. Due to this design choice, InfluxDB's aggregation capabilities could not be used and direct visualization of the data in common time-series visualization software was not possible. Time was spent assisting UDL in determining the best way to redesign this aspect of the database, making allowances to implement the new database if it had been ready during the project timeline, and finding a way to side-step the problem when it was evident that the timing of the updates to the database would not meet project deadlines. That said, the resulting design changes

should bring an improvement in performance and capabilities of UDL's InfluxDB SKYSPARK database and was a beneficial side-effect of this project.

- UDL's InfluxDB SKYSPARK database does not fully mirror all of the data from the EWS SkySpark database, it only extracts a portion of the available columns of information. The columns (tags) that had been chosen were meant to, when combined, represent a unique key that could tie the information back to a specific sensor in EWS SkySpark. However, if the value for one of those tags changed in the EWS SkySpark database, then the next time data was extracted, it effectively became a new unique sensor in the UDL InfluxDB SKYSPARK database. This lack of a proper key caused difficulties in relating queried data back to the SkySpark metadata and also back to earlier measurements in the queried data when a sensor's identifying information had changed slightly (i.e. addition or removal of the term "Pharmacy" from the groupRef tag). This has been solved by including the ID column from the SkySpark database in the database update mentioned above.

## Conclusion

The project was successful in achieving its main goal of delivering a Python program that queries data from UDL's InfluxDB mirror of the EWS SkySpark database for the UBC's Pharmacy building, cleans the data appropriately, classifies the end-use of each EC sensor and integrates everything into a Grafana dashboard. The final model was an Agglomerative clustering with Single Linkage and 20 clusters and Boosting for the supervised model. The program achieved an accuracy of 94.3% when applied to the testing dataset, categorized the 208 sensors within the Pharmacy building into end-use classes, and wrote the classifications to the END\_USE measurement in the UDL InfluxDB SKYSPARK database. A detailed report of the analysis in addition to Python code and a web-based Grafana dashboard was completed. Although the project experienced some difficulties mentioned above, we are pleased the project assisted with improving the performance of UDL's InfluxDB SKYSPARK database. We have outlined below some future work to improve the accuracy, performance, and scalability of the model. Overall, our work has helped UDL and UBC take a step closer to reaching their vision of assisted AI for proactive & preventative maintenance.

## Scope for Future Work

With the dynamic nature to the project and lack of certainty with sensor labelling there is some future work that could be done to improve the analysis of data and the model itself. Future work includes things that would increase accuracy, improve performance, and help scale the model up for other buildings.



## Accuracy Updates

Things that would help improve the accuracy of the model include:

- Determine which EC Sensors are submeters of other EC sensors to reduce or eliminate double-accounting of energy
  - Use wiring diagrams, naming conventions, etc.
  - Develop a model to classify if a meter is a main meter or a submeter
- Increase size of the hand labeled training set
  - Expanding to include other buildings
  - Use wiring diagrams, naming conventions, etc. to gain a better understanding what the unlabeled sensors do
- Modify the code to point at the updated database design when querying the sensor values (currently pointing at the `UBC\_EWS` measurement)
- Modify the code to pull the sensor metadata from the SkySpark POINTS rather than from a csv
- Investigate additional feature engineering for EC data based on power consumption curve shapes (i.e. to relate EC sensors with similar curve shapes)
- Investigate additional feature engineering for NC data such as:
  - Finding a way to represent the state of sensors that only report a string status code (may allow for more discreet clusters)
  - Converting all units to metric
  - Adding a geo-spatial aspect (may be important when expanding to work with many buildings)

## Performance Updates

Things that would help improve the performance of the model include:

- Look into if performing NC sensor aggregation as queries using Flux improves the speed of the program by reducing the required number of queries (will likely need to upgrade the UDL database to InfluxDB v1.8 so that a newer python package that supports Flux (<https://github.com/influxdata/influxdb-client-python>) can be used.
- Look into if storing non-time-series data in a lightweight SQL database enables faster querying and reduces the required storage space for the database
- General optimization of code (particularly where a series of if-statements are applied to every value in a column of a large dataframe).

## Scalability Updates

Things that would help improve the scalability of the model include:

- Make Feature Selection code dynamic (currently in a stand-alone module and the selected features are “hard-coded” into the program)
- Investigate if the query for identifying EC sensors is generalizable across all buildings or if it needs additional selection criteria.
- Update database querying functions to allow querying of multiple buildings, potential options are:
  - Implement regex in the query to allow querying multiple buildings from a string of building names
  - Loop through a list of buildings and query each individually
  - Loop through a list of buildings and dynamically add OR statements and where\_params
  - Upgrade InfluxDB to 1.8 or later and switch to using <https://github.com/influxdata/influxdb-client-python> to be able to issue Flux queries (which has a 'contains()' function).
- Confirm that all code referencing specific column names / column indices is valid for datasets from different buildings/multiple buildings.
- Adjust Grafana dashboard to allow for multiple buildings to be selected at once.

## References

- [1] J. Wei, "ABOUT," UBC Urban Data Lab. [Online]. Available: <https://urbandatalab.io/about/>. [Accessed: 01-May-2020].
- [2] J. Wei, "UBC Building Energy Data," UBC Urban Data Lab, 11-Feb-2020. [Online]. Available: <https://urbandatalab.io/news/ubc-building-energy-data/>. [Accessed: 01-May-2020].
- [3] J. Wei, "Leveraging Energy Data for Sustainable Buildings," UBC Urban Data Lab, 12-Feb-2020. [Online]. Available: <https://urbandatalab.io/project/analyzing-ubc-building-energy-use/>. [Accessed: 01-May-2020].
- [4] "Project Haystack," Tags – Project Haystack. [Online]. Available: <https://project-haystack.org/tag>. [Accessed: 01-May-2020].
- [5] X. Gao and A. Malkawi, "A new methodology for building energy performance benchmarking: An approach based on intelligent clustering algorithm," *Energy and Buildings*, vol. 84, pp. 607–616, Dec. 2014.
- [6] T. Tarpey, "Linear Transformations and the k-Means Clustering Algorithm," *The American Statistician*, vol. 61, no. 1, pp. 34–40, Jan. 2007.
- [7] "sklearn.metrics.silhouette\_score," scikit. [Online]. Available: [https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette\\_score.html](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html). [Accessed: 06-Jun-2020].

## Glossary

<b>Accuracy</b>	The proportion of predictions the model got right.
<b>Feature Selection</b>	The process of reducing the number of predictor variables. Some of the top reasons to use feature selection is it enables models to train faster and reduces the complexity of models for easier interpretation.
<b>F1 Score</b>	The balance between precision and recall.
<b>Log Loss</b>	It measures the performance of a classification model where the prediction input is a probability value between 0 and 1. The goal of machine learning models is to minimize this value. A perfect model would have a log loss of 0. Log loss increases as the predicted probability diverges from the actual label.
<b>Precision</b>	The proportion of positive identifications that were actually correct.
<b>Recall</b>	The proportion of actual positives that were identified correctly.
<b>Silhouette Score</b>	<p>A measure of how similar an object is to its own cluster compared to other clusters. Guidelines on how to interpret the silhouette score are [7]:</p> <ul style="list-style-type: none"><li>• 1: Identified clusters are distinct with no overlap</li><li>• 0: Clusters overlap</li><li>• -1: Observations have been assigned to incorrect clusters</li></ul>

## Appendix A: Detailed Flowchart

A more detailed way of viewing the project flow is by organizing it by the following tasks:

- I. Data Collection and Cleaning
- II. Feature Selection and Engineering
- III. End-Use Classification
- IV. Pipeline Outputs
- V. Visualization

These five steps are shown below in **Figure 9** overlaid with detailed components of the project showing the flow of data from the initial query all the way through to the dashboard visualization. Black arrows represent the general flow of data, and the red and green arrows show the specific EC and NC sensor data flow, respectively. The data starts in UDL InfluxDB SKYSPARK database and also in the original EWS SkySpark database, passing into the *Data Collection and Cleaning* phase, outlined in the purple box. Next, the data goes into the *Feature Selection and Engineering* phase, outlined in the green box, where only the important features are kept. The next phase is the *End-Use Classification* phase, in the yellow box, where the data is fed into our model and end-use categories are predicted. After that, the phase in the blue box is *Pipeline Outputs*, where the results are stored back in the UDL InfluxDB SKYSPARK database. The last phase is to use the stored end-use labels to create a web-based Grafana dashboard that has a graph showing energy consumption for a building, which is shown in the red box.

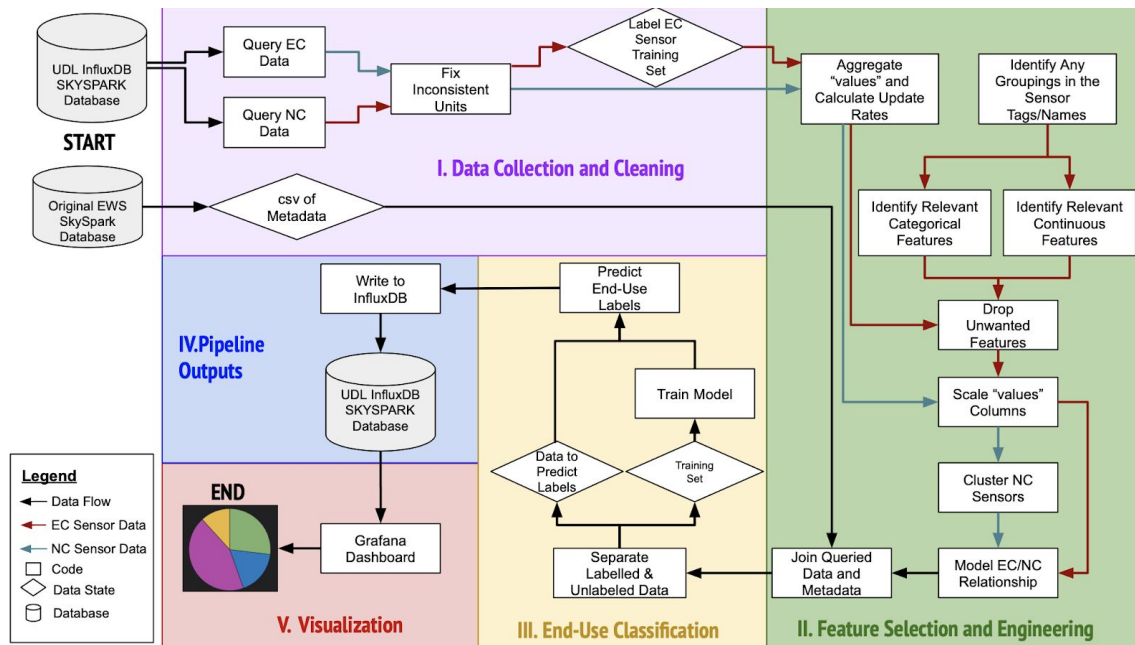


Figure 9: Function Flow

## Appendix B: Clustering Methodology

### Gower's Distance

The clustering on NC sensors was performed on both continuous and categorical data, as a result Gower's distance was employed in order to calculate the relative distance between observations due to its ability to account for mixed data types. The continuous data that was considered is listed below:

- Mean sensor measurement
- Minimum sensor measurement
- Maximum sensor measurement
- Standard deviation of the sensor measurements
- Mean sensor update rate per day

The categorical data being considered was the sensor's unit of measurement (°C, kPa, etc...). The unit of measurement column was encoded prior to the Gower's distance calculation since such computations only accept numeric input values.

### Multidimensional Scaling

Gower's distance is a method of quantifying the relative distance between each observation. The output of the Gower's distance calculation is an  $n \times n$  matrix, where  $n$  is the number of observations. Many clustering methods are able to accept pre-computed distances, for these methods the Gower's distance can be passed into the model directly. However, some of the clustering methods that were considered were unable to accept Gower's distances and expected data in a predictor space to be passed in.

Since the data to be clustered had mixed data types, it was not appropriate to immediately pass the raw data into models that require a predictor space as input. In order to account for this, Multidimensional Scaling (MDS) was applied to the Gower's distance matrix. MDS is a method of converting relative distances between observations into a coordinate space. It is capable of applying this conversion to any number of dimensions, meaning that it can also be used as a form of feature reduction without significant loss of information. In this case, the data was scaled down to two dimensions for the following reasons:

- The number of dimensions has minimal impact on the results of the clustering methods since the relative distances are maintained
- Visualization of clusters is simplest in two dimensions

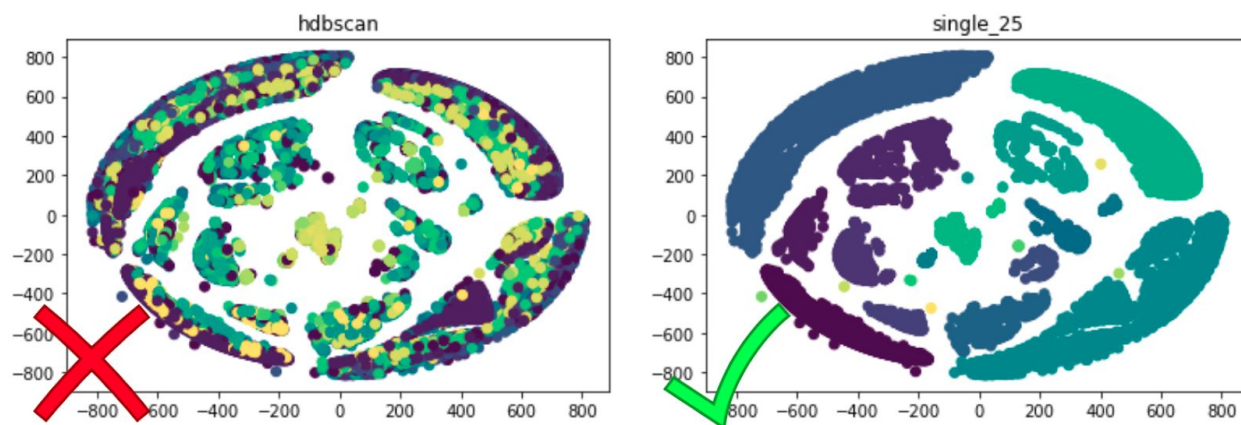


### Methods of Comparing Clustering Models

Various different clustering methods and configurations were tested, the silhouette scores and plots of the clusters were used in order to identify which clustering methods did an adequate job of identifying the clusters within the data.

The silhouette score is a relatively objective method of comparing the effectiveness of a clustering model. More notably, visually interpreting plots of clusters is significantly more subjective in many cases as it requires the use of judgement. This method of comparing clustering models is useful as model comparison metrics can behave erratically when clusters have variable size and shape. Visual interpretations of plots can be used to identify erratic behavior.

From visual interpretations of plots it was clear that some clustering methods performed better on the data than others. For example, density based clustering methods where the model identifies the number of clusters on its own, such as HDBSCAN, had trouble identifying unique clusters within this data, as shown on the left side of **Figure 10**. However, agglomerative methods appeared to do a relatively good job of identifying unique clusters within the data as shown on the right side of **Figure 10** below.



**Figure 10:** Examples of good and bad clustering (Left: HDBSCAN, Right: 25 Cluster Single Linkage Agglomerative)

### Clustering Model Selection

Various clustering methods and configurations were tested. Some methods required a user-defined number of clusters, where this was the case a grid search was employed to find the optimal number, testing from 5 through 55 clusters. Other methods required a user-defined linkage method; where this was the case another grid search was employed to additionally test each linkage method with each number of clusters (5 through 55) to identify the optimal parameters for each configuration.

The silhouette scores and plots of clusters resulting from the various grid searches were compared and seven clustering configurations were selected to be tested as possible

feature engineering methods for the final model. The clustering method that produced the best prediction results when applied to the testing data was then implemented in the final version of the model.

**Table 5** below shows the optimal configurations of the different clustering models and configurations considered, along with if it was selected to be tested as possible feature selection methods for the final model, and the reason for whether or not it was selected.

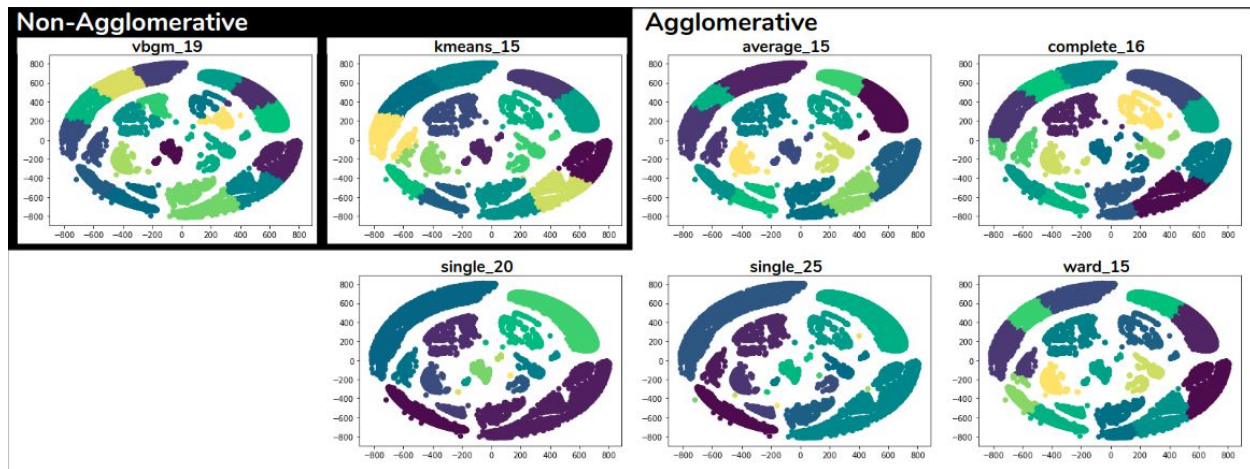
**Table 5:** Clustering Model Comparison

Model	Optimal Number of Clusters	Linkage	Selected For Comparison	Reason
<b>Agglomerative</b>	15	Average	Yes	- Reasonable silhouette score - Clear visible delineation of plotted central clusters
<b>Agglomerative</b>	16	Complete	Yes	- Reasonable silhouette score - Clear visible delineation of plotted central clusters
<b>Agglomerative</b>	20	Single	Yes	- Best visual identification of plotted clusters
<b>Agglomerative</b>	25	Single	Yes	- Best visual identification of plotted clusters
<b>Agglomerative</b>	15	Ward	Yes	- Reasonable silhouette score - Clear visible delineation of plotted central clusters
<b>DBSCAN</b>	-	-	No	- Poor silhouette score - Plotted clusters not representative of visible groupings
<b>Fuzzy C-Means</b>	35	-	No	- Relatively low silhouette score - Plotted clusters not representative of visible groupings
<b>Gaussian Mixture Models</b>	25	-	No	- Unable to generate sufficiently small groupings of plotted clusters within central clusters at a reasonable number of clusters
<b>HDBSCAN</b>	-	-	No	- Poor silhouette score - Plotted clusters not representative of visible groupings
<b>K-Means</b>	15	-	Yes	- Reasonable silhouette score - Clear visible delineation of plotted central clusters
<b>Meanshift</b>	-	-	No	- Poor silhouette score - Plotted clusters not representative of visible groupings
<b>Variational Bayesian Estimation of a Gaussian Mixture</b>	19	-	Yes	- Best silhouette score - Clear visible delineation of plotted central clusters

The seven clustering models considered for final implementation were broken out into two categories as listed below:

- Non-agglomerative:
  - Variational Bayesian estimation of a Gaussian mixture (vbgm\_19)
  - K-means (kmeans\_15)
- Agglomerative with Different Linkages:
  - Average (average\_15)
  - Complete (complete\_15)
  - Single (single\_20 and single\_25)
  - Ward (ward\_15)

Plots of each of the above listed clustering methods are shown in **Figure 11** below.



**Figure 11:** Plots of the Clustering Methods Considered for the Final Model

As mentioned in **Step 1) Cluster NC Data** the agglomerative clustering models with single linkage had the lowest silhouette scores but provided the best cluster definition when observed graphically. This inconsistency from what is expected in the agglomerative clustering model with single linkage stems from the nature and shape of the clusters appearing within this data. The long, thin, curved clusters can be more readily identified by the single linkage method than other methods as this method of clustering creates clusters through the following procedure:

1. Assign each observation to its own cluster
2. Calculate distances between each cluster using the shortest distance between the clusters
3. Join the two clusters with the smallest relative distance between each other
4. Repeat steps 2 and 3 until the desired number of clusters have been

The nature of the single linkage is that the shortest distance between clusters is used to calculate the distance between clusters in step 2. In contrast, the average linkage model calculates the distance between clusters in step 2 as the average distance between

observations within the clusters. This nature inherent to the single linkage distance calculation allows irregularly shaped clusters such as the long, thin, and curved clusters that can be seen encircling the other clusters in **Figure 11** to be identified as individual clusters. This phenomenon can be seen in the case of the two single linkage models shown in **Figure 11** above, while the same perimeter clusters tend to be broken into several different clusters when applied to the other five clustering methods that achieved higher silhouette scores. These higher silhouette scores stem from how the silhouette score is calculated when using the Scikit-Learn `silhouette_score` function from the metrics package. The silhouette score returned from the `silhouette_score` function is the average of the Silhouette Coefficients of each observation which can be represented by the following equation [7]:

$$\text{Silhouette Score} = \sum_{i=1}^n \text{Silhouette Coefficient}_i$$

Where n is the number of observations within the dataset being clustered. The Silhouette Coefficients are calculated for each observation using the following equation [7]:

$$\text{Silhouette Coefficient} = \frac{(b-a)}{\max(a, b)}$$

Where a is the mean intra-cluster (or within cluster) distance while b is the mean nearest-cluster distance (or distance from the observation to the nearest cluster) [7].