Energy Modeling at Cornell

Steve Mandl
Department of Building Automation and Control Systems Integration, Cornell University

---

Overview

This paper describes the techniques used in developing the Utilities Metering and Modeling Application (UMMA) for predicting utility consumption on the Ithaca campus. There are approximately 1,800 meters which are currently being modeled by this system. I will describe the machine learning (ML) methods used to build predictive models, analysis used to determine the appropriate ML processes, interactive visualization techniques deployed to facilitate information retrieval. After discussion of the current state of the system, there is some discussion about related systems and potential improvements to the system.

Application Description

UMMA in its current state is driven by the Cornell University Infrastructure, Properties, and Planning associate director of budget, along with a small team of finance specialists and utilities engineers. The platform for the main application uses a Javascript web UI, with a node.js application server, Oracle SQL database, and python ML model generation & pre-processing. Other systems report out the data maintained within UMMA in dashboards to reach a wider audience.

At the time this was developed, python was chosen as the language to be used for learning the models, because there was not yet a good ML library for Javascript. Scikit-learn is the standard du jour library for ML in python. It is well established and supports a robust collection of ML algorithms and pre-processing utilities. (Scikit-learn) Now that several javascript libraries are available and supported (Chang, Node-svm), the python functionality may be ported to Javascript so the server component can run purely in node.js. This abridgement should simplify deployment and maintenance.

The UMMA application is designed to assist in the analysis of Utility (metered energy) consumption at Cornell University. The energy metering data is managed by the EBS application, and UMMA shares the same database as EBS. Meters are read approximately on a monthly basis. An actual meter reading contains the following attributes:

- Period denotes the year and month (YYYYMM) of the end of the interval
- Meter ID is unique for each meter. If a meter gets replaced, the new meter may have the same meter ID as the old meter. The physical representation of what energy the meter is measuring may also change for a given meter ID. For example, if an Energy Conservation Initiative (ECI) project is implemented in a facility, the consumption before and after this project may look quite different.
- Start Date represents the start of the interval energy consumption is measured for.
- End Date represents the end of the interval energy consumption is measured for.
- Actual Consumption is the number of units of energy consumption this meter recorded during the interval
- HDD is the Heating Degree Days during the interval.

- CDD is the Cooling degree Days during the interval.[1]

In addition to analyzing historic consumption data, UMMA also can model consumption as a function of weather for an individual meter. UMMA will generate (train) models for a single meter or for all meters. Once generated, these models can then be used to predict future consumption and this is used as a basis for financial budgets for energy usage. The models can be used to project expected consumption for a given meter, and given environmental conditions (actual or forecast weather and date components) for the period being predicted. The model is trained on a sample of data from a date range specified by the user. This date range must include at least 12 consumption readings for different periods for each model being trained. Models also have auxiliary attributes such as:

- The netid of the creator (user requesting model creation)
- Notes specified by the creator, useful for documenting the reason for the model creation or what application the model is intended to be used for
- The date range used for training
- The initial accuracy score as compared to the training data
- The training sample size
- A user specified multiplier that can be used for tuning a model in anticipation of deviations from historical consumption trends.
- Active flag. Since multiple models can exist for a given meter, models have a flag specifying whether they are active or not. Business rules ensure that only the most recent model for a meter is active. The notion of an active model is the best one available for current meter and facility conditions, and is the one used for generating future predictions that will inform financial budgeting.
- Minimum value. When the model is trained, the lowest actual consumption value from the training data is recorded in this field. Predictions generated by this model are constrained to never be less than this minimum value.[2]

There are sometimes situations where the historic consumption readings for a meter are sporadic, not representative of expected future consumption, or there are less than 12 actual readings yet because the meter is too new. In these situations, an accurate model cannot be trained based on historical data. To facilitate these situations, models can be synthesized by using a model template that predicts the average consumption 'shape' for a given commodity and unit of measure. The model synthesis scales a model template to target a yearly consumption specified by the user. In this way, models can be built for all meters with any specified consumption budget.

Projected consumption values can be generated for a given fiscal year (presumably the upcoming one). This will generate predicted consumption values for the 12 periods within the fiscal year, based on the forecast weather data and the active model for any meter included in this export. Models can be re-trained or scaled using the multiplier, then different projected consumption values can be generated.

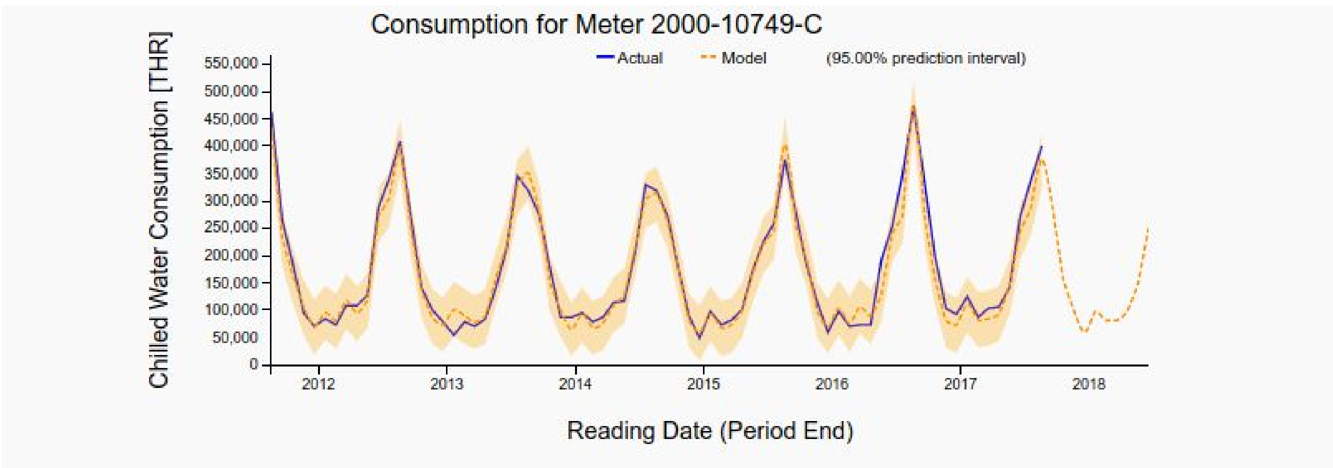
---

[1] Degree Days are computed using daily averages with a 65 °F base.

[2] With the deployment of distributed production facilities such as rooftop solar arrays, production meters as well as consumption meters are likely to make their way into this system. The convention in utilities systems is to report production as negative consumption, so any assumptions about meter readings reporting positive numbers will need to be adjusted.

This iterative supervised validation process allows for tuning of the models. Once the set of active models represents the best estimate as determined by a human expert, the utilities budget records become locked by an administrator.

UMMA allows analysis of a single model, graphing actual consumption and predicted consumption on the same axes. This graph is accompanied by actual and predicted consumption in the various aggregations as well as the statistics Prediction_Error and R^2 correlation (see figure 1). This analysis will be computed for the entire date range of data available for the meter, or for a smaller range specified by the user. A smaller range can be specified by 'brushing' the graph with the mouse or a touchscreen.



Analysis for Periods billed 2011-08-20 to 2018-06-20

Chilled Water Consumption [THR]

Total Actual Consumption:	13.0211M	**Total Model Consumption:**	13.7831M
Actual Consumption Per Day:	6.05069k	**Model Consumption Per Day:**	5.63495k
Actual Consumption Per HDD:	330.736	**Model Consumption Per HDD:**	299.989
Actual Consumption Per CDD:	4.18415k	**Model Consumption Per CDD:**	4.17561k

Total Days:	2.44600k	**Total CDD:**	3.30086k	**Total HDD:**	45.9453k

Total Enthalpy Hours Dry air 0°F Basis (BTUH/lb -):	1.12253M
Avg. Enthalpy over Period (BTU/lb):	19.1219
Actual/Model % Difference	-5.53%
R^2	95.84%

Figure 1

UMMA allows similar analysis to the single meter analysis, but for the totals for groups of meters within a building or in groups of buildings where meters can be combined with the same commodity and unit of measure. Building performance can be analyzed by graphically representing a building's performance measured as the actual/model percentage difference for a given date range and commodity. This will allow performance comparison of multiple buildings at a glance.

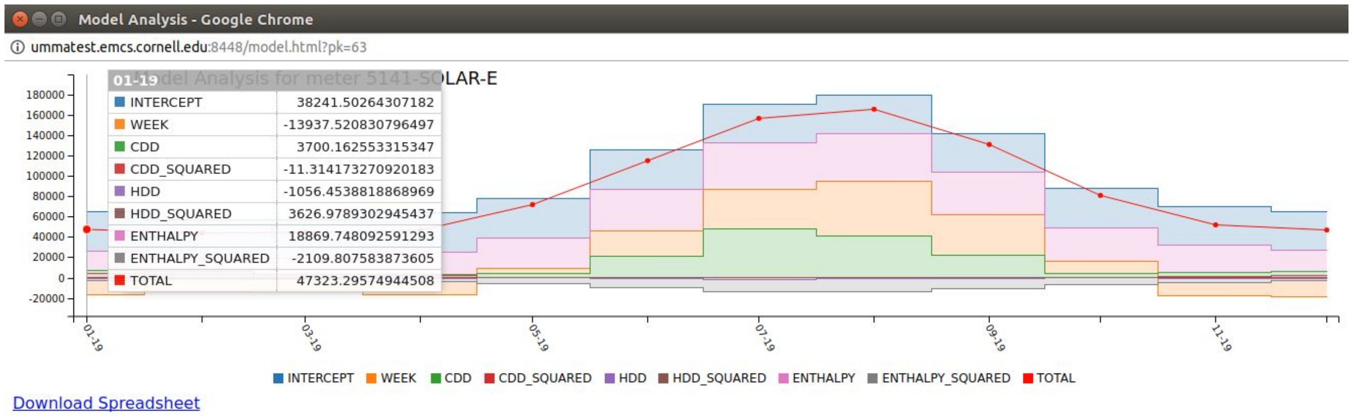


Figure 2. Model component analysis

Model analysis for a single model, as in figure 2, is available to help understand how different features are weighted comparatively within the model. A raw weather chart as displayed in figure 3 shows daily weather values which all models' training and predictions are based on. Past values are based on actual observed weather, and future values are seven year average predictions.

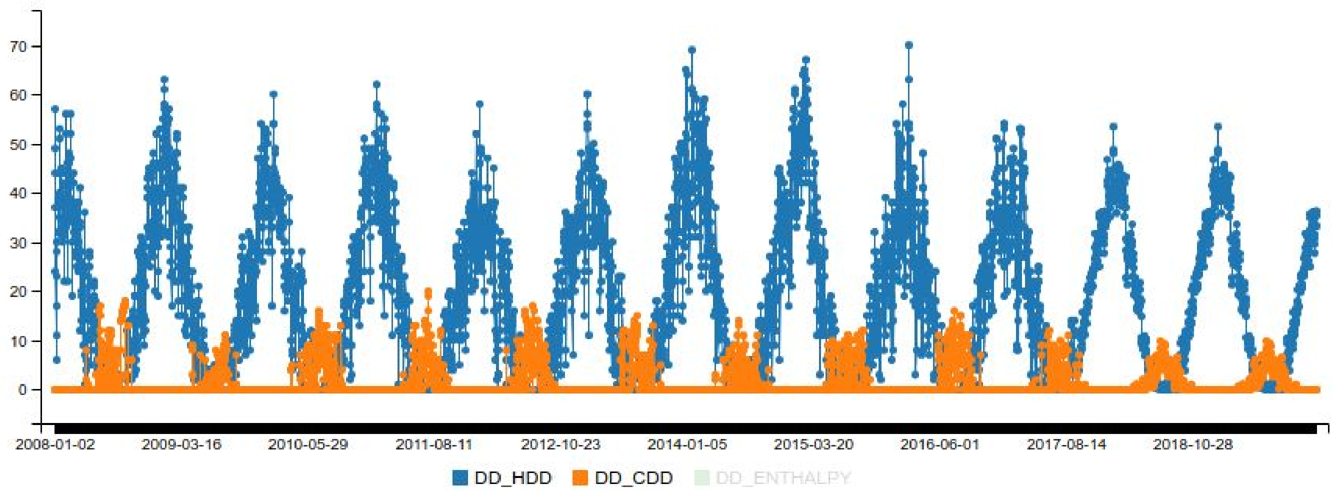


Figure 3

Development of Application

Meters are installed in campus facilities to measure energy delivered over time. The underlying assumption is that the energy measured corresponds to the energy consumed within that period of time. There are some cases where this assumption does not always hold: For example, a few facilities are heated with fuel oil stored in a tank nearby. The metering of these is accomplished when the fuel is delivered, and the amount delivered can vary depending on the inventory levels prior to delivery, as well as the whim of the vendor. The same meter will typically stay in place across transitions of the underlying system consuming the energy. For example, an electric meter may be installed for a building entry point, and if the building gets LED lighting installed, the same meter may persist. In cases like this, it is important to realize that modeling consumption before and after the implementation of such a project is a different task; models can be used to compare consumption before and after, but if the model strives to minimize prediction error, the same model will not be accurate across the system change.

Model Creation

When model creation is requested, the user specifies a date range to use for training data, and a set of meters to build models for. The process creates separate parameterized models for each of the meters specified using available historical consumption data from the requested date range and corresponding to each meter. For each meter, the process applies the epsilon Support Vector Regression (SVR) algorithm. The parameters ε and c are incorporated into the objective function (equation 1)applied to the training data x:

Equation 1

$$\text{Minimize} \qquad \frac{1}{2}\|w\|^2 + c\sum_{i=1}^{l}(\xi_i + \xi_i^*)$$

$$\text{Subject to} \qquad \begin{cases} y_i - \langle w, x_i \rangle - b \leq \epsilon + \xi_i \\ \langle w, x_i \rangle + b - y_i \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases}$$

Optimal SVM parameters ε and C are found by minimizing RMS error on between modeled consumption and actual validation consumption using 3-fold cross validation to avoid overfitting. The decision to use regression SVM to compute models was based on several criteria: 1 - Typically the sample size is small (20-60) compared to the feature space (~60 features). 2 - Least squares regression will typically overfit with comparably small sample sizes. 3 - Finding the optimum model build parameters is relatively easy because the parameter space is only two dimensional.

To avoid overfitting, SVR is used while adjusting the C and ε parameters with resampling (cross validation). This method is computationally expensive, but with such small sample sizes, the performance is manageable. Manually adjusting these parameters for every model is not practical since a large number (up to 1800) of models need to be created in batches.

The optimization of SVR parameters C and ε is accomplished by maximizing the average r^2 (coefficient of determination) of predicted consumption against validation data over 3-fold randomly selected cross validation subsets of the sample of actual consumption. Figure 4 visualizes how the predictive performance of a model performs as a function of the (C, ε) SVR parameters. Due to the complex nature of this surface, finding the optimal SVR parameters using gradient descent may not always converge on the optimal parameters due to getting stuck in local maxima.[3] Instead, a brute-force 'shotgun' approach is used searching over the entire grid of parameter combinations to find the optimum.

Once the model $\widehat{f}$ is computed, the model is stored as a weight vector w of the SVR, including a y intercept b. Additionally, there is a scalar multiplier feature m which facilitates manual scaling of a

[3] A process such as simulated annealing could apply to finding the optimal SVM parameters, and may be an interesting area of future investigation.

model by a prescribed multiple. In order to then use a model to compute consumption for a period, the system only needs to calculate the following:

$$y = m(w^{\top} x + b) . \qquad \textbf{equation 2}$$

Since x is pre-processed in the database, and m, w, and b are all stored in the database model representation, computing the output of a model for any period can be done simply by evaluating the arithmetic expression in equation 2.

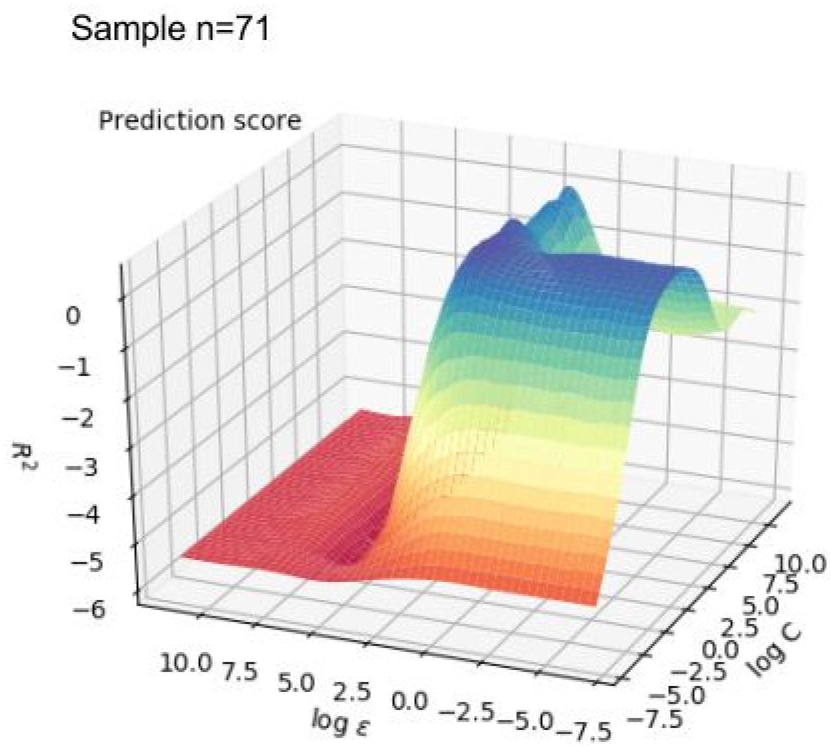


Figure 4 Potable water meter prediction scores considered for optimal model parameters

Selection of ML algorithms

One of the use cases of the UMMA system is to allow a utility engineer to create models on particular meters based on user specified date ranges. Once a model is built, the user will evaluate the model's output, and perhaps refine the model for various reasons. This is important because systemic changes can affect the consumption function of the underlying environmental data. For example, when an Energy Conservation Initiative was implemented on a building such as the 2014 Mann Library ECI project (Paradise), a model based on consumption data prior to the project was no longer be accurate when applied to time periods after the change. In this case, the engineer would select a shorter time period for the training sample starting after the change in order to better predict current and future consumption in the system being metered. Other less dramatic changes such as installing LED lights in a building may affect the consumption enough to warrant building a new model on post-change data, or else adjusting a model with a percentage multiplier if not enough post-change training data is available.

This use case requires that single models need to be built in a time period on the order of a few seconds or less, which is no problem on modern computers especially with small training sample sizes.

Another aspect of the use case involves quickly generating output from the predictive models, and making this function available not only through the application interface, but also to reporting tools that support ad-hoc inquiries. Accomplishing this was done by allowing the pre-processing of observed (and forecast) environmental data and transforming it into the feature space required for a model to digest. In order to facilitate repeatable model prediction across various systems, a rich enough feature space was developed to allow an accurate and informed linear regression to closely fit the training data. Once the input x_i is calculated in the feature space, a model can be represented by a vector *w*, and the output of the model is easily computed as the dot product $w^T x_i$.

A novel aspect of the UMMA system is the way features are pre-processed in the database. This removes the necessity of using a kernel function for the SVM during training or when predicting consumption values. This pre-processing occurs identically for actual and projected points in the feature space. Some of these features are nonlinear with respect to the raw weather parameters, so the models can fit nonlinear time series data while only needing to learn a linear regression.

Feature Selection

Prior to the development of UMMA, consumption predictions were done manually using a physics based model. For heat and cooling consumption, the rate of consumption is strongly dependent on degree days. Cooling also more closely tracks enthalpy than cooling degree days, since humidity affects the ability for the cooling system to condition air. Figure 5 illustrates the relationship between enthalpy and a typical chilled water consumption in a building with conditioned air. While there is a clear dependence on enthalpy, the r^2 values usually are below 0.3 even in the best cases.

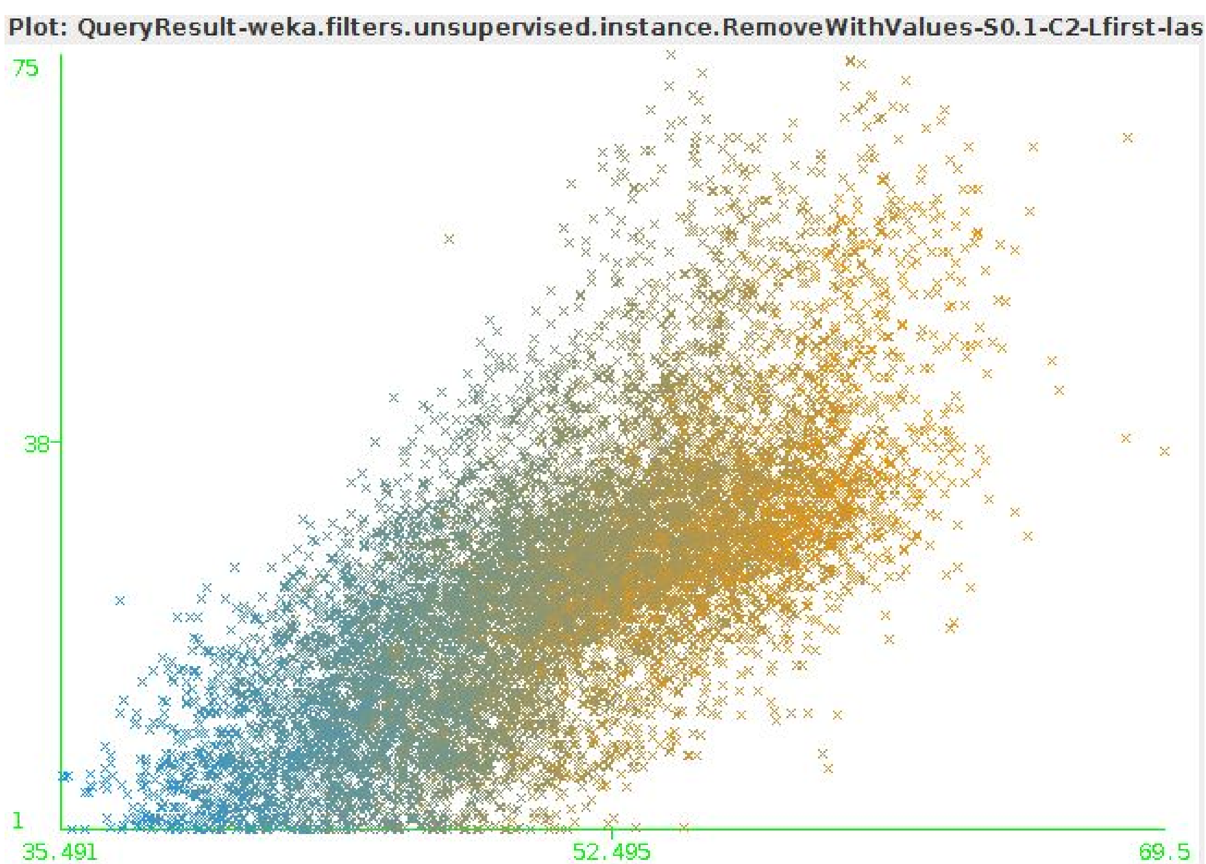


figure 5 Daily CW consumption vs. enthalpy in Alice Cook Hall 2015-2017

Correlation between some interesting features is depicted in figure 6. This illustration shows samples from three meters, each with a different color. The bottom row represents consumption, and the bottom right is consumption correlated with enthalpy. The two meters colored in blue exhibit good correlation between consumption and enthalpy, however the red meter exhibits a pathology that would plague a model which was based primarily on enthalpy. Since the building underlying the red meter had programs that were seasonally dependent, the consumption followed one trend when it was primarily occupied, and a significantly different trend when it was primarily unoccupied. This seasonal dependence is widespread, and varies with space usage. A laboratory building with fume hoods often open during classes should exhibit a spike in both heating and cooling during the semesters. Other space types such as residence halls, research space, or administrative offices will exhibit different seasonal profiles.

Another observation with consumption vs. enthalpy is the fit is curved, not linear. In order to best fit data like in this situation even without the time dependent consumption (as in the red meter) the model will need to fit a polynomial curve with degree greater than 1. It is feasible to accomplish this nonlinearity via preprocessing since there are only a few (HDD, CDD, and Enthalpy) features that are real numbers with a polynomial relationship to consumption. Coupled with the fact that polynomial of order 2 is adequate to give a good fit, there is little processing or memory cost with the preprocessing necessary to fit the observed nonlinearity. This simpler option of preprocessing obviates the need to employ a nonlinear kernel function. While Smola and Schölkopf advocated using a kernel in instances of high dimensionality or high polynomial degree, neither of those cases apply here. (Smola)

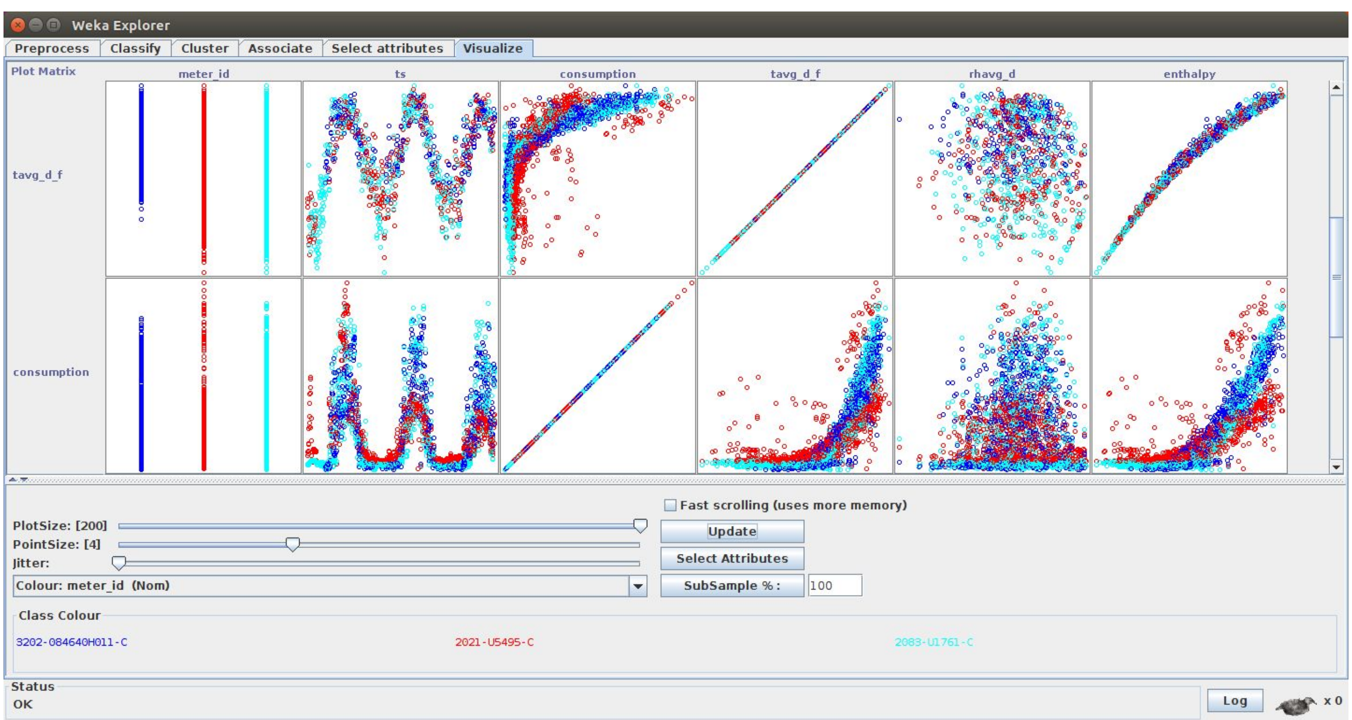


Figure 6

The feature space consists of the following tuples, where the degree days and enthalpy values are sums of daily values for all days in the period, and a period is defined by the interval from the period's startdate to the enddate (timestamps). The week features are as follows: $week_i$ represents the *i*th week of the year. The value of $week_i$ is based on *v*, the number of days of overlap between the period's interval and the days in the *i*th week. Since $(0 \leq v \leq 7)$, and $week_i = v/7$, $(0 \leq week_i \leq 1)$. The feature vector $(HDD,\ CDD,\ Enthalpy,\ HDD^2,\ CDD^2,\ Enthalpy^2, week_1,\ week_2,\ ...,\ week_{52})$ contains features expressing the degree of overlap between the billing period and the week of the year. All 52 week features are computed for each period. These features are normalized and pre-processed in the database that stores the historic actual consumption readings, as well as future records which contain forecast weather data and period start/stop times.

Analysis and Presentation of Model Output

The main visualization of model output is a time series chart that overlays actual consumption with modeled consumption computed using equation 2. The modeled consumption is bracketed with a band whose width represents a 95% prediction interval. The prediction interval *pint* is computed by taking the 95th percentile of the absolute value of the model versus actual. Let y_i be the actual consumption at period $i\,(1 \leq i \leq n)$, x_n is the multivariate input based on actual weather and environmental observations, and $\widehat{f}_c(x_n)$ is the output of a prediction function learned from training set *c* applied to multivariate input x_n.

$$pint = P_{95}[abs(y_i - \widehat{f}_c(x_i))]_1^n$$ **Equation 3**

The method of estimating the prediction interval described in equation 3 was chosen in lieu of in-sample prediction error or apparent error. (Borra) There is some optimism built into the formula in equation 3 due to applying the loss function to data that includes the training set. However, the method is tolerant to skewed extra-sample errors that are often observed in the metering data. This is not the case in the techniques described by Borra and Di Ciaccio, which assume the error observed is normally distributed with zero mean. Such formulas incorporating summations of the error tend to be overly pessimistic if the mean of the error distribution wanders too far from zero.

The model in figure 1 was trained on a sample size n=59 on a date range from 2011-07 to 2016-06. The prediction interval estimated with equation 3 is displayed as a dark orange band around the modeled consumption.

In order to give some insight into the components that make up the predictive nature of a model, the visualization in figure 2 shows how the model would predict consumption for a year with average environmental data. The consumption along with the contribution from each of the dimensions in the feature space is graphed to allow deeper analysis. This allows insight into how much of consumption for a given month is due to weather driven features, and how much is due to time of year (weather independent) features. This type of inquiry could help inform decisions based on projections for climate change, or conversely, predicting changes to consumption in the event of a significant change to the academic schedule.

Related Systems and Potential Improvements

Predicted weather is based on the average of a 7 year sliding window of the historic weather for each day of the year. The weather for a day consists of a vector of daily average temperature (degrees fahrenheit) t_avg_f , daily average relative humidity (percent) rh_avg, enthalpy of moist air according to the ASHRAE psychrometric formula, Heating Degree Days = greatest(0, 65 - t_avg_f), Cooling Degree Days = greatest(0, t_avg_f - 65). The day number *d* of some future date is calculated (1 to 365, or 366 in leap years), then the mean of the vectors of the corresponding day from each year in the sliding window represents a future day's forecast weather.

The workflow of UMMA includes running the prediction for a calendar year at a time, typically 2-3 years ahead of time. Once this is complete, the models are generated for the list of meters expected to be installed during the upcoming fiscal year. These models are validated by comparing their output to previous years and evaluating expected changes known by the facilities staff due to systemic changes. Models can be adjusted using a multiplier for anticipated structural changes. Once the models all pass validation, they are frozen and consumption values are applied to billing rates to feed into the financial budget. These budgeted expenses are split according to space usage, and the utilities budget for all departments gets published.

Periodically throughout the year as the meters are read, the actual consumption is compared with weather-adjusted model predictions to indicate weather adjusted building performance compared to the budget baseline.

Currently models are built for financial budgeting. A monthly prediction of energy cost is predicted based on the model's projected consumption times the billing rate for the commodity. After converting energy consumption to BTU's, and coupled with net area served by an individual meter, the Energy Use Intensity (EUI) can be calculated: Total yearly BTU / Net Area. The EUI is a standard indicator of building performance, and is useful in comparing performance of Cornell facilities to facilities of similar types at other institutions or industry averages

A potential improvement would involve utilizing automated meter reads available through smart networked meters. Much higher resolution than the monthly meter reads utilized in UMMA are available, and may be useful in many cases. The accuracy of the UMMA models could be improved by adding smaller time windows along with the corresponding larger training sample sizes. Some interesting future work would involve measuring how the prediction accuracy improves as the measurement window shrinks.

Additional features to feed into the ML algorithm(s) could also become available by accessing data from the integrated Energy Management Control System, such as: building occupancy, air flow rates, setpoints, lighting levels, etc… In addition to potentially improving prediction accuracy, other applications may become available with models operating on higher resolution data, such as:

- Predicting impacts of short term (around 1 week or less) events such as supply failures, controlled load shedding, and scheduled plant maintenance. By using more accurate short term weather forecasts as input to a consumption model, a short term demand profile could be generated. Coupled with production schedules, risk of supply shortages could be calculated and used to inform production decisions. A separate analysis of expected model predictive error for models based on short term weather forecasts would need to be performed in order to understand the risk of shortages.

- Understanding potential impacts of changing campus work schedules, semester start and end dates, and building controls standards. Projections of impact to energy use due to modifying setpoints or changes to occupancy rates due to schedule or policy changes could be useful in evaluating decisions about these. Even small changes like the effect of posting signs around the building could be measured if the measurements were sensitive enough and the models could separate out the ambient noise of normal consumption. With daily measurement of building energy, feedback to the building occupants could be made available almost immediately, perhaps soon enough that they could remember what changes they made had a measurable effect.

Acknowledgements

I would like to thank Dr. Carla Gomes for her discussions and support in writing this paper. The research and development are part of an ongoing project within the department of Energy and Sustainability at Cornell University.

Works Cited

Borra, Simone, and Agostino Di Ciaccio. “Measuring the Prediction Error. A Comparison of Cross-Validation, Bootstrap and Covariance Penalty Methods.” Computational Statistics & Data Analysis, vol. 54, no. 12, 2010, pp. 2976–2989., doi:10.1016/j.csda.2010.03.004.

Chang, Chih-Chung and Lin, Chih-Jen. “LIBSVM: a library for support vector machines.” ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

Cherkassky, Vladimir, and Yunqian Ma. “Practical Selection of SVM Parameters and Noise Estimation for SVM Regression.” Neural Networks, vol. 17, no. 1, 2004, pp. 113–126., doi:10.1016/s0893-6080(03)00169-2.

Node-svm: Support Vector Machines for npm-js, http://www.npmjs.com/package/node-svm.

Paradise, Peter. “Energy Conservation Initiative Project Summary, Mann Library.” energyandsustainability.fs.cornell.edu/file/Mann.pdf.

Scikit-Learn: Machine Learning in Python, http://scikit-learn.org/stable/.

Smola, Alex J., and Bernhard Schölkopf. “A Tutorial on Support Vector Regression.” Statistics and Computing, vol. 14, no. 3, 2004, pp. 199–222., doi:10.1023/b:stco.0000035301.49549.88.