



Anomaly Detection with Energy Data

Team:

Fei Wang
Yumeng Ding
Gautam Moogimane


Sponsor:

Jones Lang LaSalle Americas, Inc. (JLL)
Linnea Paton



Executive Summary

- Jones Lang LaSalle Americas, Inc. (JLL) requested support to automate the process of detecting anomalies in their energy consumption.
- Team tried multiple methods to automate detection of anomalies in the data including SARIMA, Holt-Winter, and Neural Networks.
- Three methods that were finalized and presented fast/consistent results were- STL Decomposition, Prophet, and Clustering.



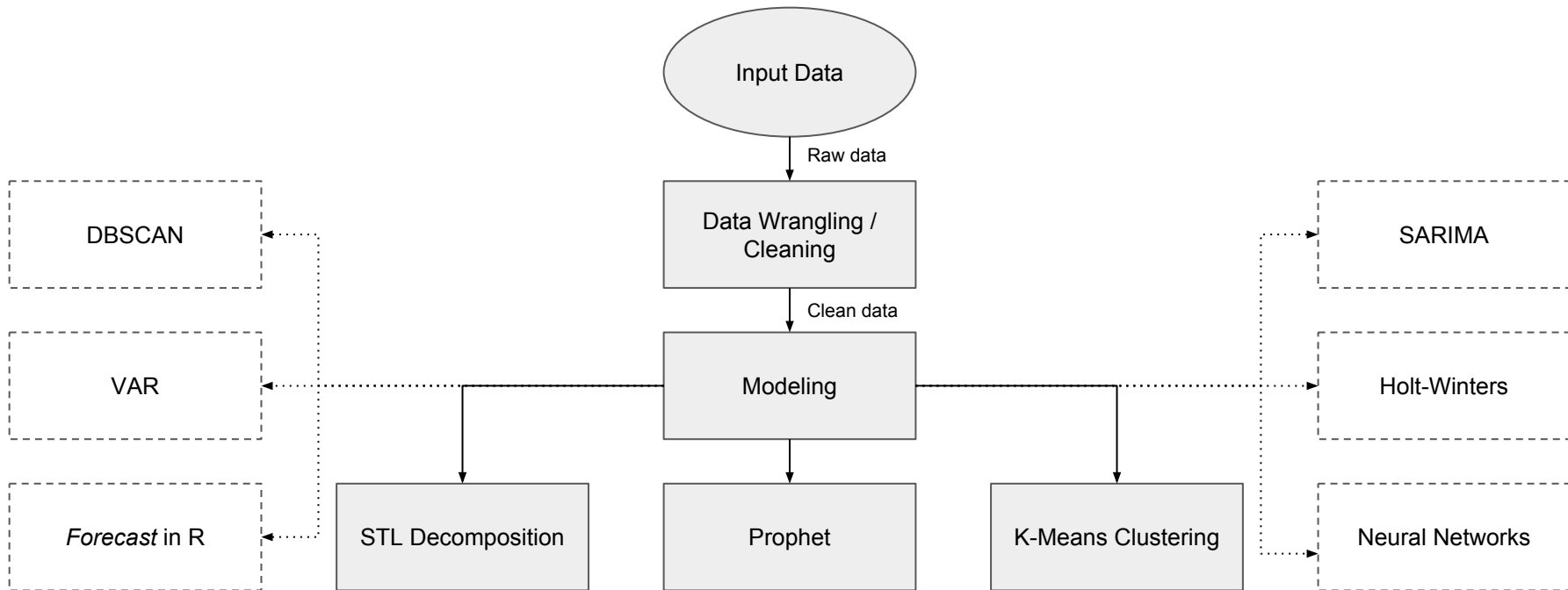
Helping clients reduce energy usage to have an accretive impact to P&L

Background

- Energy consumption is increasing every year making it crucial to effectively monitor usage, so as to conserve resources.
- Anomalous energy data can be reported due to a number of reasons like malfunctioning equipment, faulty construction or even manual entry errors.
- Identifying actual anomalies from spikes and drops due to known factors such as seasonality, can be a challenge.
- Current approach used by the client is a combination of manual eyeballing and use of a multivariate linear regression model called PEERS.

Our Approach

To develop automated, quick and accurate models that can identify anomalies in reported energy consumption.





Deliverables

- Data wrangling
 - Provide an automated process to detect missing data at energy account level
- Outlier detection procedures
 - Automated, quick procedures to detect outlier values at energy account level
 - Scalable to thousands of accounts



Data Wrangling & Cleaning

- NYCHA public dataset:
 - Electricity Consumption Bill data from 2009 to 2018
 - 313,147 rows, 27 attributes
 - Attributes: Building_ID, Meter_Number, KWH_Consumption, Charges
- Consolidation of Account_ID's
 - Separated meter numbers for KWH and KW consumptions
 - Switches of meter number over years under the same accounts
 - Typos in meter numbers
 - ~3,000 accounts with 9 years' of monthly electricity consumption data



Data Wrangling & Cleaning

- Proration of KWH Consumption to calendar months
- Imputation of KWH Consumptions

Building_Meter	Revenue_Month	Service_Start_Date	Service_End_Date	Consumption_KWH
341.0 - BLD 04_7835072	2012-03-01	2012-02-24	2012-03-26	13760.0
341.0 - BLD 04_7835072	2012-04-01	2012-03-26	2012-04-24	11600.0
341.0 - BLD 04_7835072	2012-05-01	2012-04-24	2012-05-23	10160.0
341.0 - BLD 04_7835072	2012-06-01	2012-05-23	2012-06-22	10240.0
341.0 - BLD 04_7835072	2012-09-01	2012-08-22	2012-09-21	12320.0
341.0 - BLD 04_7835072	2012-10-01	2012-09-21	2012-10-23	12800.0

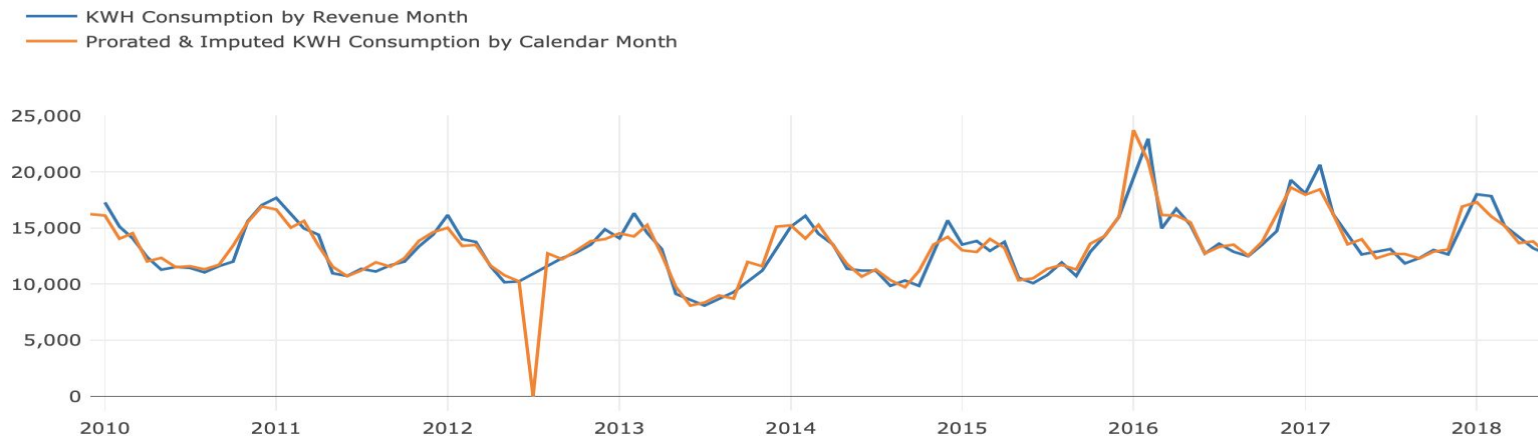


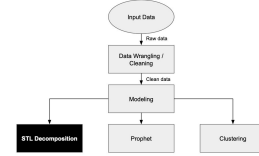
Building_Meter	Calendar_Month	Imputed_KWH
341.0 - BLD 04_7835072	2012-03-01	13496.774194
341.0 - BLD 04_7835072	2012-04-01	11652.413793
341.0 - BLD 04_7835072	2012-05-01	10779.586207
341.0 - BLD 04_7835072	2012-06-01	10240.000000
341.0 - BLD 04_7835072	2012-07-01	0.000000
341.0 - BLD 04_7835072	2012-08-01	12730.666667
341.0 - BLD 04_7835072	2012-09-01	12213.333333
341.0 - BLD 04_7835072	2012-10-01	12995.862069



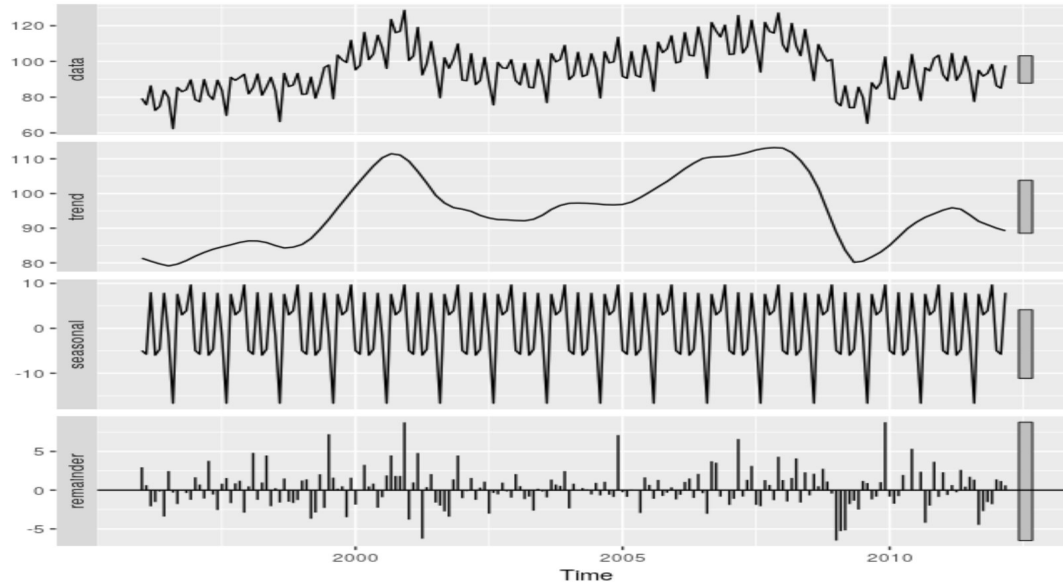
Data Wrangling & Cleaning

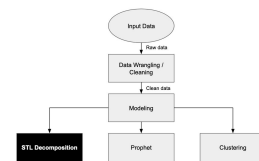
KWH Consumptions over time





Method 1 - Time Series Decomposition

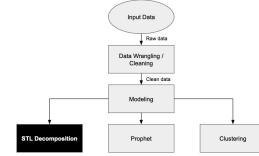




Method 1 - Time Series Decomposition

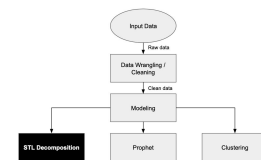
STL: Seasonal and Trend decomposition using Loess

- Split the time series into 3 components: Trend, Seasonal, Residual
 - Cleveland et al. (1990)
- Why STL
 - Robust to outliers - anomalous values will not affect the estimate of trend and seasonal components, but will remain in the residual component
 - Seasonal component is allowed to change over time (suitable for energy consumption pattern changes over the years)

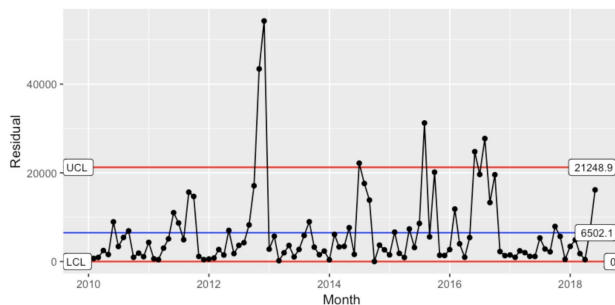
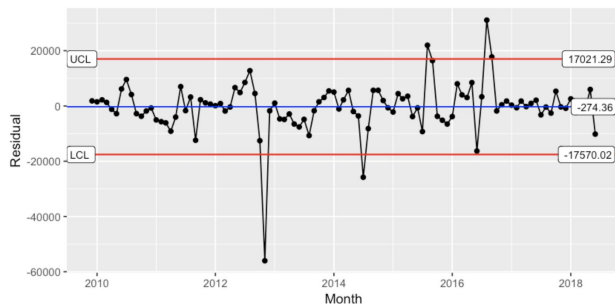


Detecting anomalies from residuals

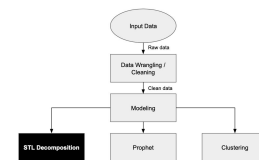
- Detect anomalies from the residuals
 - XmR chart
 - Interquartile Range
 - Generalized ESD Test (not applicable)



Detecting anomalies from residuals

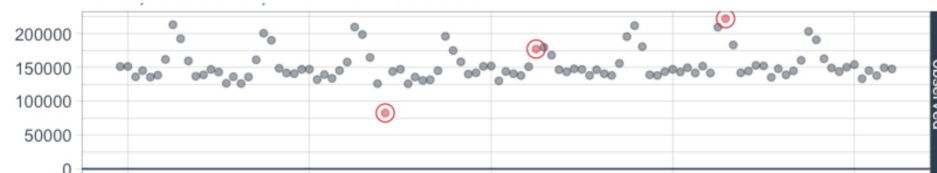
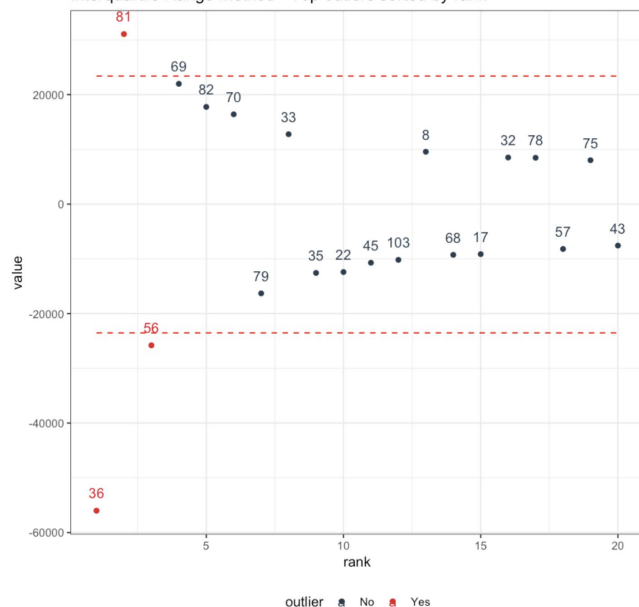


Month	Total	Trend	Seasonal	Residual
2012-11-01	82710.3	150984.3	-12265.344	-56008.64
2016-08-01	222030.0	156279.1	34686.263	31064.60
2014-07-01	177073.8	153031.5	49841.165	-25798.86
2015-08-01	211696.2	155032.3	34686.263	21977.61
2016-09-01	183134.8	156329.0	9046.662	17759.19

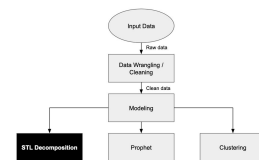


Detecting anomalies from residuals

Interquartile Range Method - Top outliers sorted by rank

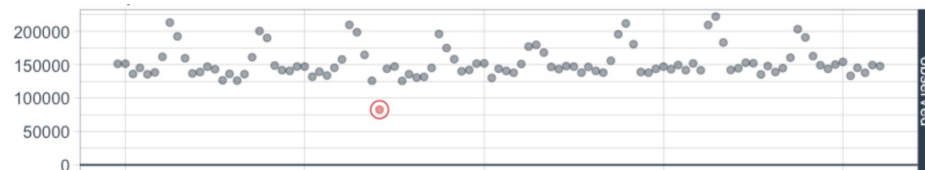
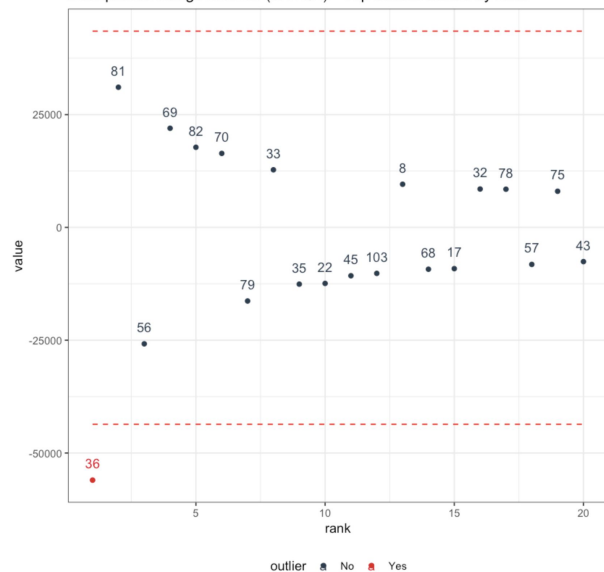


Building_Meter	Calendar_Month	Total	Trend	Seasonal	Residual
165.0 - BLD 03_90327795	2012-11-01	82710.3	150984.3	-12265.34	-56008.64
165.0 - BLD 03_90327795	2016-08-01	222030.0	156279.1	34686.26	31064.60
165.0 - BLD 03_90327795	2014-07-01	177073.8	153031.5	49841.16	-25798.86

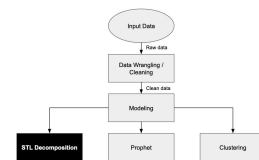


Detecting anomalies from residuals

Interquartile Range Method (6X IQR) - Top outliers sorted by rank



Building_Meter	Calendar_Month	Total	Trend	Seasonal	Residual
165.0 - BLD 03_90327795	2012-11-01	82710.3	150984.3	-12265.34	-56008.64

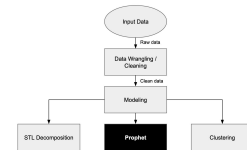


Results

3 outliers detected from the example account (ordered by a weighted rank from the 3 methods):

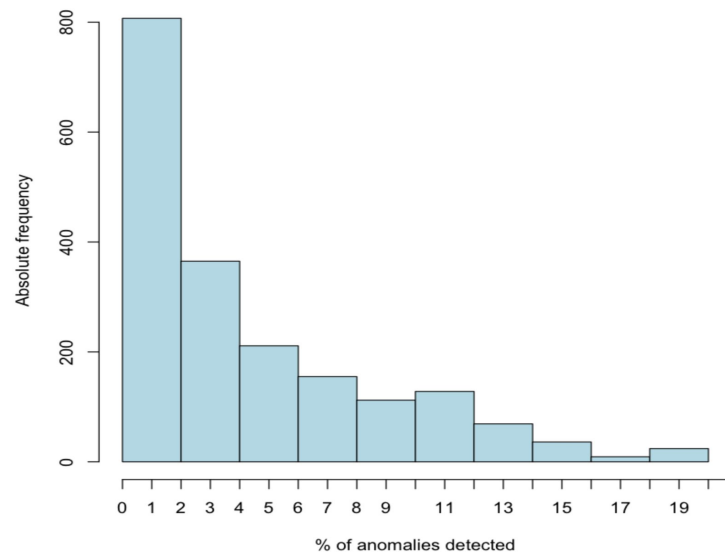
Building_Meter	Calendar_Month	Total	Trend	Seasonal	Residual
165.0 - BLD 03_90327795	2012-11-01	82710.3	150984.3	-12265.34	-56008.64
165.0 - BLD 03_90327795	2016-08-01	222030.0	156279.1	34686.26	31064.60
165.0 - BLD 03_90327795	2014-07-01	177073.8	153031.5	49841.16	-25798.86

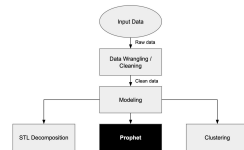
Runtime: ~0.1 seconds



Summary of outliers detected

Frequency count of % of anomalies, Decomposition



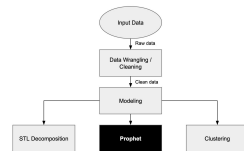


Method 2 - Prophet

Open source decomposable additive regression time series model developed by Facebook's core data science group with 4 main components:

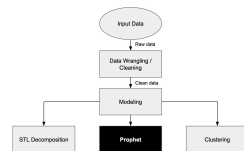
- Linear/Logistic growth trend curve - $g(t)$
- Weekly/Yearly seasonality component modeled using Fourier series - $s(t)$
- User defined list of holidays - $h(t)$
- Residual/Error - which are assumed to be i.i.d and normally distributed - e

$$y(t) \leftarrow g(t) + s(t) + h(t) + e$$



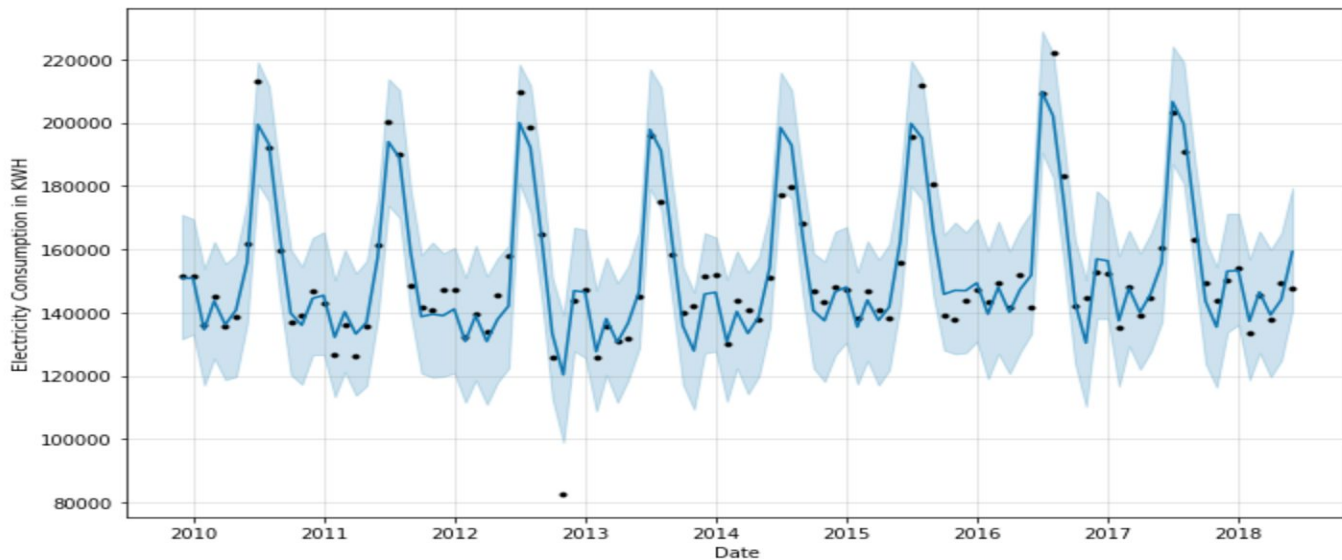
Why Prophet

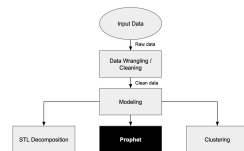
- Intuitive parameters that are easy to tune.
- Fast and fairly accurate forecasting that can be automated.
- Automatically detects changes in trend by selecting changepoints in the data.
- Output dataframe contains uncertainty intervals in addition to forecasted values, for the entire dataset, to aid in outlier detection.
- Unlike other popular time series forecasting models, it is robust to missing data , and outliers do not adversely impact the predictions.
- One of the few available forecasting libraries available in Python, with the majority released in R.



Detecting Anomalies using Prophet

Using Monte Carlo simulations, Prophet generates uncertainty intervals for the forecast, with an upper and lower bound.



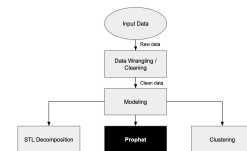


Results

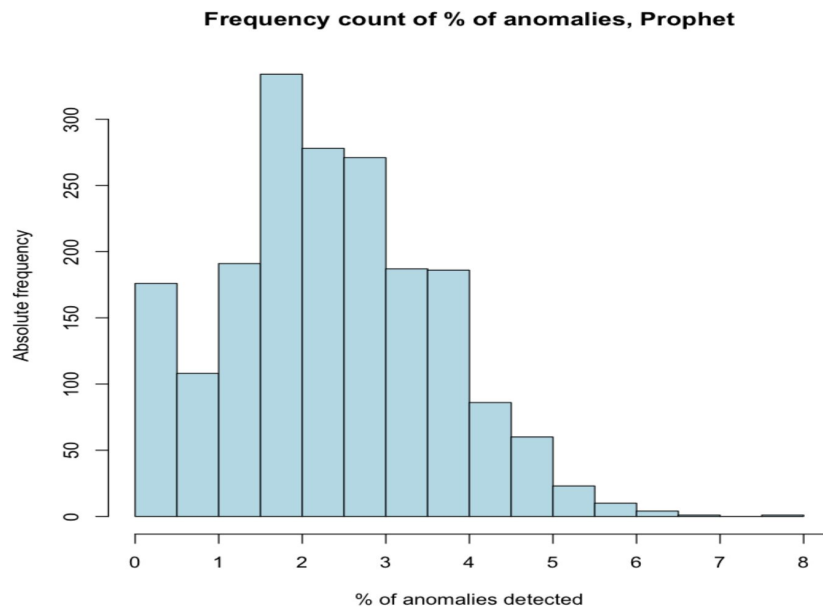
Anomalies identified for the same sample account, after tuning the model

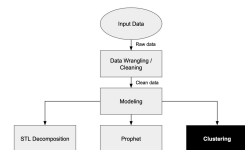
	ds	yhat	yhat_lower	yhat_upper	y_orig	Anomaly
35	2012-11-01	120337.617440	98863.694986	139086.916304	82710.303030	Yes
55	2014-07-01	198516.935479	179158.960448	215947.573832	177073.777778	Yes
80	2016-08-01	202179.846799	182634.272355	221958.055563	222030.000000	Yes

Runtime: ~0.1 seconds



Summary of outliers detected

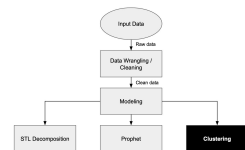




Method 3 - Time Series Clustering

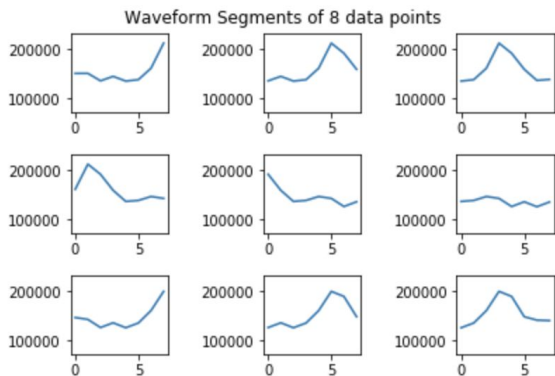
Time Series Clustering: Unsupervised K-Means clustering from sklearn.cluster package

- Segment time series trends into waveforms of 8 data point each
- Perform K-Means clustering using squared euclidean distance to find nearest centroids
- Reconstruct time series trend using nearest centroids for each data points
- Detect anomalies from analysing reconstruction error larger than certain threshold

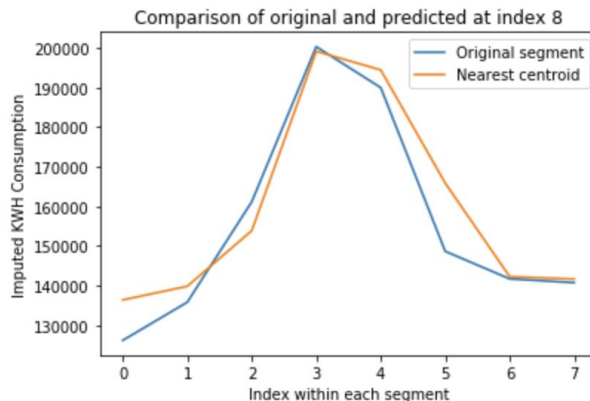


Clustering Process

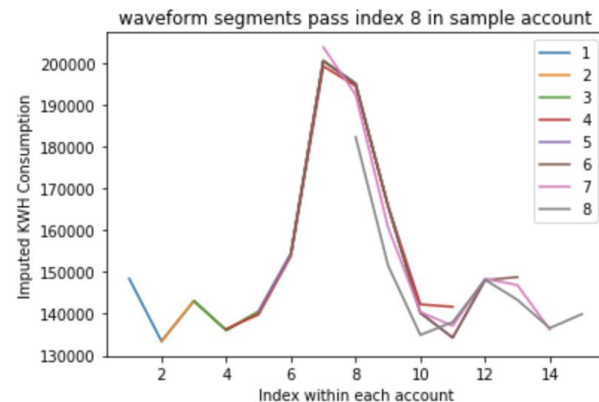
Step 1 - segment time series trend into waveforms and perform K-means clustering

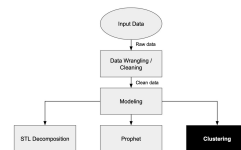


Step 2 - for each segment, find the nearest centroid from K-means clustering



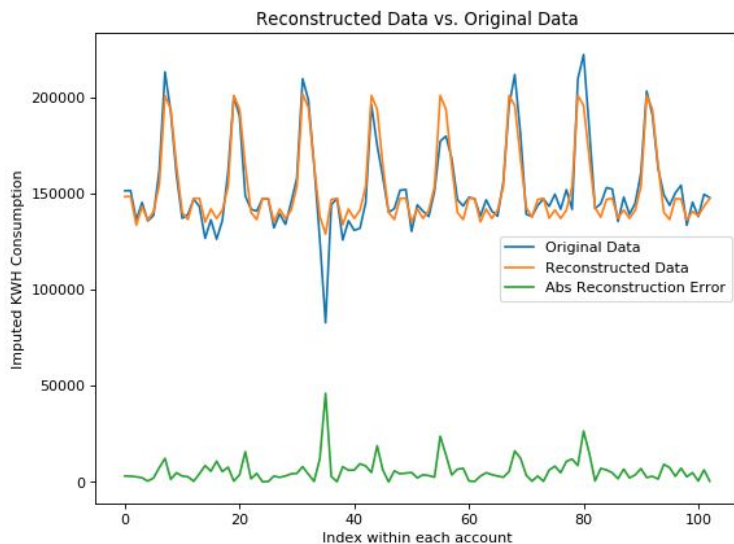
Step 3 - Combine nearest centroids for each data point to generate prediction



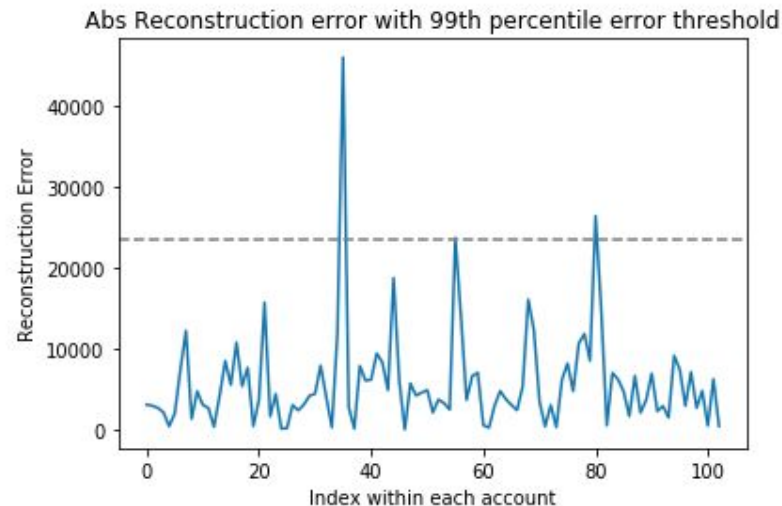


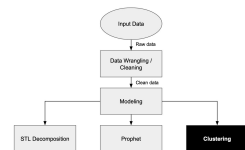
Detecting Anomalies by Reconstruction

Reconstructing the time series trend using nearest centroids for each data point



Detecting anomaly by identifying reconstruction error larger than 99th percentile of absolute error



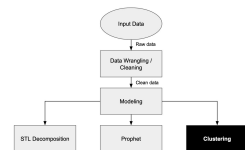


Results

3 Outliers detected from the same example account:

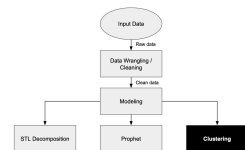
	Building_Meter	Month	Imputed_KWH	Reconstructed Value	Error	Anomaly
35	165.0 - BLD 03_90327795	2012-11-01	82710.303030	100566.872043	17856.569013	Yes
55	165.0 - BLD 03_90327795	2014-07-01	177073.777778	199587.469937	22513.692159	Yes
80	165.0 - BLD 03_90327795	2016-08-01	222030.000000	202305.276261	-19724.723739	Yes

Runtime: ~0.22 seconds per account

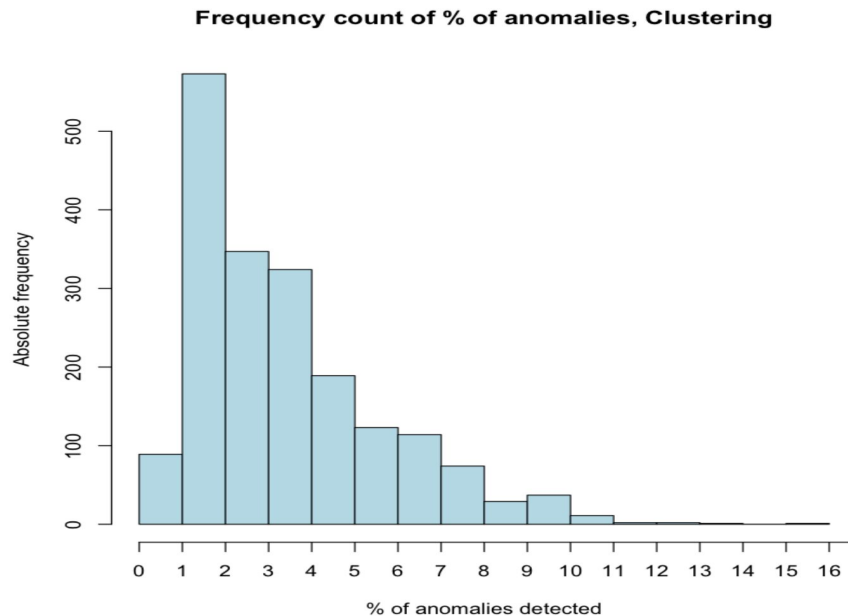


Why Time Series Clustering

- **Pros:**
 - Unsupervised learning method to detect anomaly in time series data with the ability to automatically detect different scale anomalies across multiple accounts.
 - More interpretable results and adjustable parameters to control number of anomaly points returned
- **Con:**
 - Not enough data points to accurately cluster waveforms



Summary of outliers detected





Comparison of methods

	Runtime	Interpretability	Robustness to missing values	Easiness to implement
Decomposition	Low (~0.1s)	High	High	Low
Prophet	Low	Low	High	High
Clustering	Medium (~0.2s)	High	Medium	Medium



Schedule Review

WEEK	ORIGINAL SCHEDULE	UPDATED SCHEDULE	PROGRESS
Winter Break	Proposal Revision and Data Finalization	No Change	Complete
Week 1 - 2	Create Evaluation Metrics and Evaluate Previously Established Rules	Benchmark Historic Anomaly Rate; Understand and Evaluate Established Client System	Complete
Week 3 - 8	Create 5 Anomaly Detection Methods and Evaluate Newly Established Rules	Created 3 methods in parallel and cross-validated across 3 methods	In Progress
Week 9 - 10	Review Rules with Sponsor and Prepare Poster and Docs	No Change	On Track



Current Challenges

- Lack of real client data
 - On average 2 years' of utility bill data per account
- How to evaluate model accuracy
- Decide on how to set the threshold for anomaly detection
 - Auto-tuning the parameter



Next Steps

- Methods Evaluation with Sponsor
 - Robustness to missing values
 - Short time series data
- Poster & Final Report



Q&A