Classifying User Affluence

Using power consumption data to derive customer ACORN classification.

About The Project

Our objectives.

Use Energy Consumption Data to Determine the “Affluence” of a Customer.

- Affluence is determined by membership in the ACORN group “Affluent”
- Machine learning techniques are used to associate a customer with the classification of “Affluent” or “Not-Affluent” based on consumption data.
- In particular we focus on support vector machine style classifiers.

Background & Initial Data Prep.

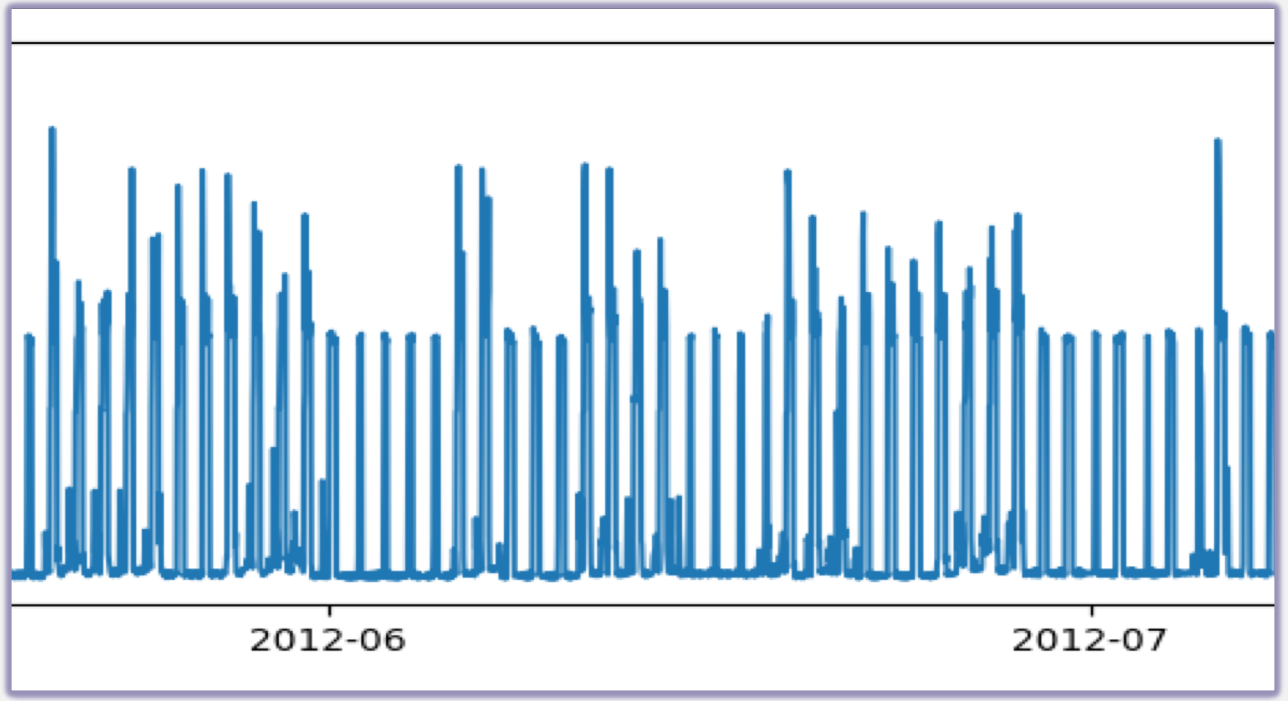
What kind of power usage data we have, where it came from, how we prepared it, and

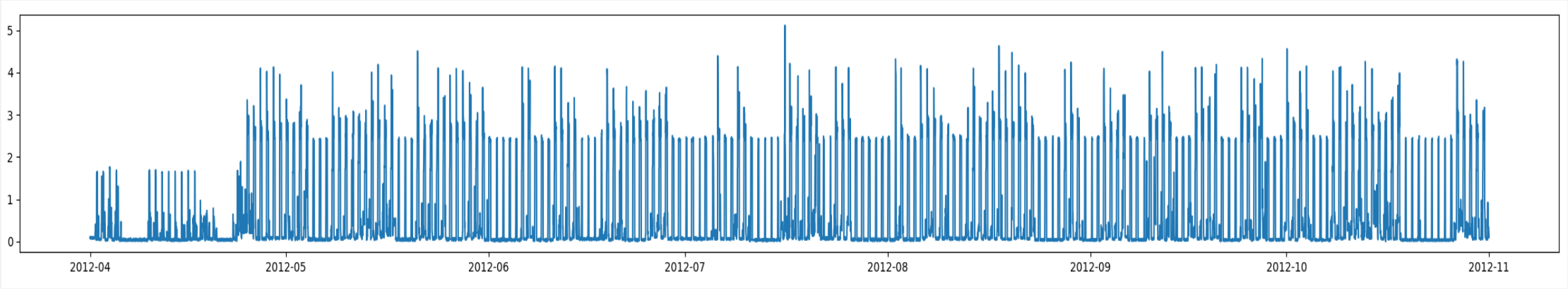
Background

Energy Consumption Data For London Households

- The data we are using is provided by the UK government.
- Raw energy consumption readings for about 5,567 London Households which took part in the UK Power Networks led Low Carbon London project
- Readings from between November 2011 and February 2014.
- About 10GB of data.
- Each sample usage profile present has an ACORN classification associated with the readings.

What do these usage profiles look like?





Example Consumption Profile

Issues With The Data

There are some issues with the data we need to discuss.

- NaN and “0” values.
  - A large portion of the usage profiles have NaN values (on the order of 60% or more)
  - The smart-meters only record values with an accuracy of about 0.001 kWh; meaning that the readings are already binned by rounding.
  - Due to the rounding, the value of “0” is over represented.
- Years with readings
  - The sample is from a pool of customers which opt-in to having their data recorded
  - While 5,567 households are present in the sample, they do not all have valid readings for the same dates

    As an example, customers drop out of the study before the end date, while others don’t start until it’s almost over.

    (this is where all the NaNs came from)
  - The year with the most complete records is 2013.
- Duplicate Records

Cleaning of The Data

Thresholding, dropping duplicates, and sample scoring.

- Duplicate Records
  - Used `pandas.DataFrame.drop_duplicates`
- Thresholding
  - Restricted to sample profiles with fewer than 10% of their data being either NaN or “0”.
- Remaining NaN values
  - Tried a complicated Markov-Chain model to replace NaNs (this did not work.)
  - Tried replacing NaN with mean customer usage (this was not representative)
  - Tried replacing NaN values with “0” (compounded the issue with “0” being over represented)
  - **Left them in place.**

Sample & Model Selection

Feature extraction, sample selection, and model selection.

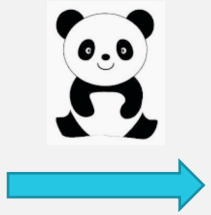
Feature Extraction

Daily Totals

	id	2013-01-01	2013-01-02	2013-01-03	2013-01-04	.....	2013-12-28	2013-12-29	2013-12-30	2013-12-31	acorn label
0	2	10.800	13.300	10.074	9.857		13.535	14.876	13.924	13.415000	Affluent
1	6	0.500	0.496	0.494	0.495		2.887	3.047	3.862	3.261000	Adversity
2	10	32.330	32.559	30.772	33.313	.....	45.916	38.192	41.568	37.697998	Comfortable
3	18	11.789	14.018	14.675	17.501		11.750	14.956	14.143	9.387000	Affluent
4	19	8.924	8.421	7.818	7.549		8.111	8.894	7.193	8.410000	Adversity
5	27	13.953	8.885	7.428	9.140		12.628	8.647	8.208	8.727000	Comfortable

```
model.fit(df, label)
```

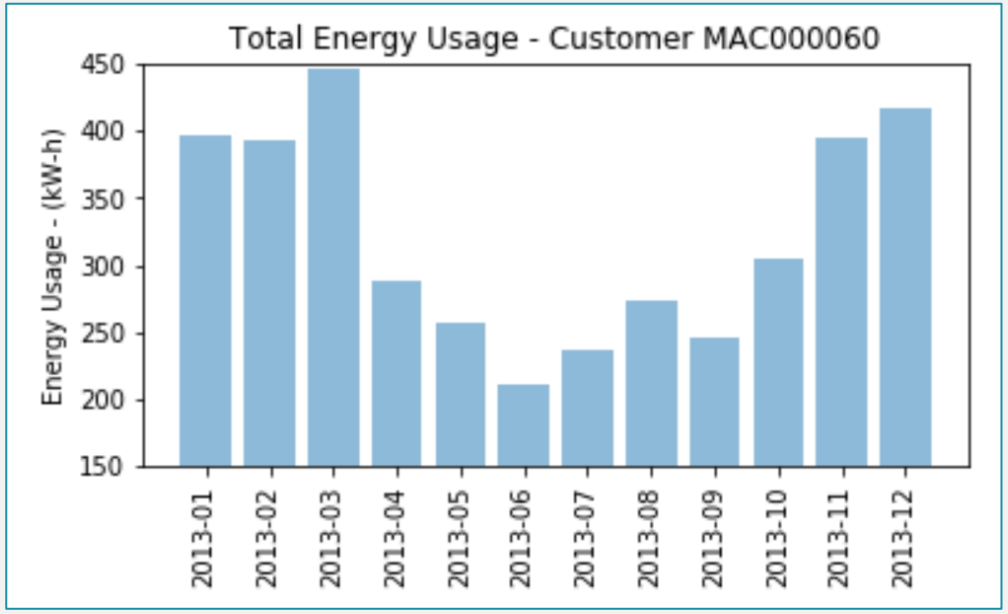
Energy reading every hour of 2013 (17520 readings) for each customer (4411).



Daily total energy consumption per customer (365).

Refined Feature Extraction

Monthly Totals

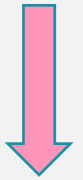


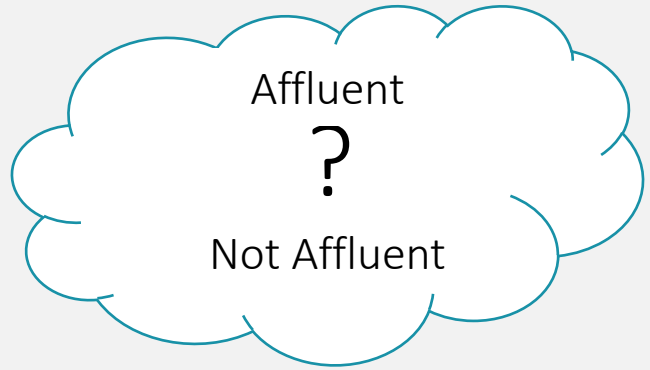
Identify ‘perfect’ customers

- Customers with a complete, Nan-less record of energy consumption every half-hour throughought 2013.
- Later question: How does a model that is successful with the ‘perfect’ customers generalize to the entire set.
- 881 perfect customers / 4411 total customers (<10% Nan)

Monthly Aggregation

- Total energy consumption over the month.
- Maximum reported energy consumption.
- Mean value.
- Standard deviation – is daily variability important ?





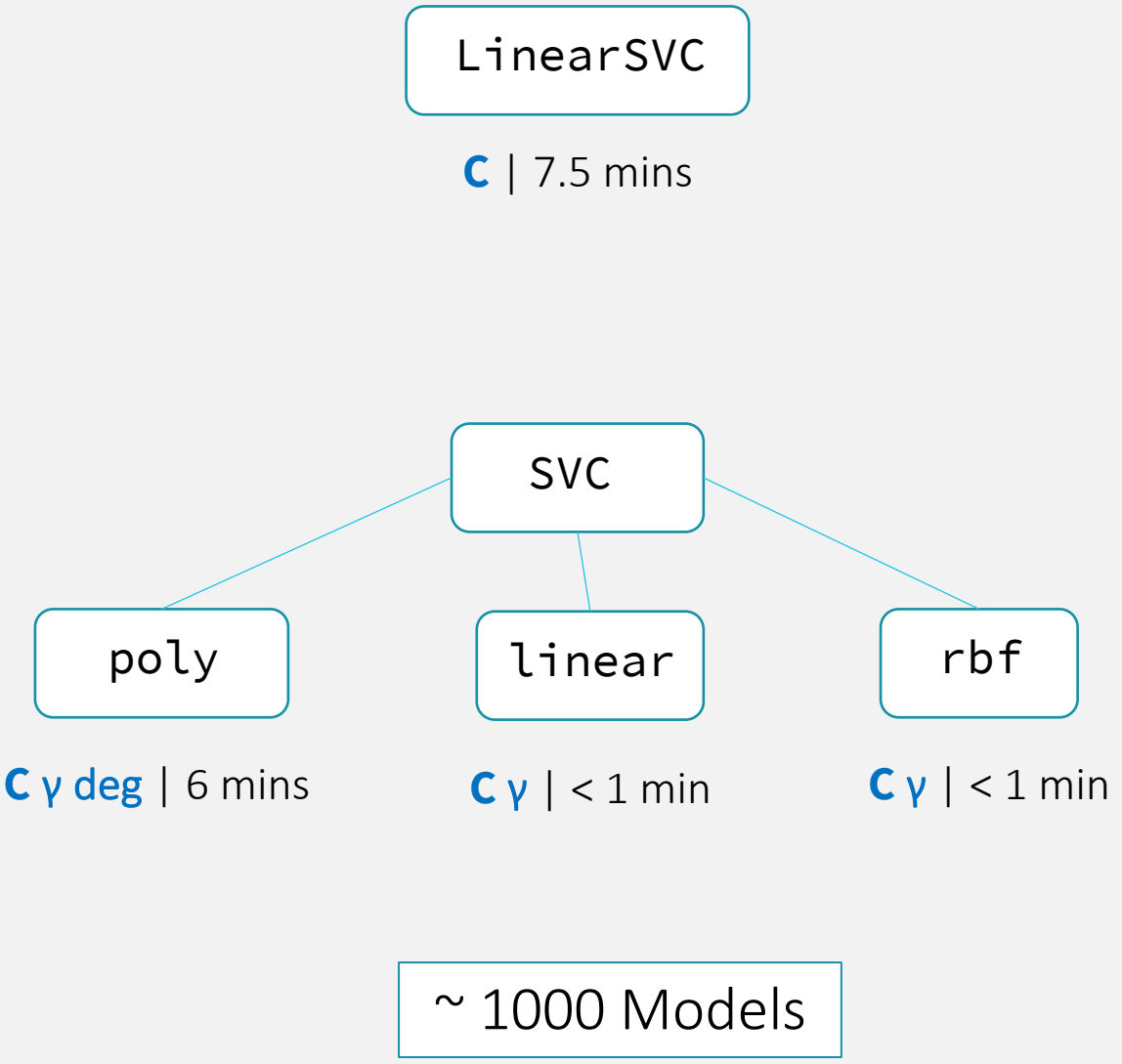
Model Selection

Single Class Classification

- SVM (Support Vector Machine) - `sklearn`
  - LinearSVC
  - SVC - Linear
  - SVC - Radial Basis Function
  - SVC - Polynomial

- Parameter Tuning - `GridSearchCV`
  - Cross Validation
  - C-parameter - [1, 2, 5, 10, 20, 50]
  - γ-parameter - [.0001, .0005, 0.001, .005, .01]
  - Deg-parameter (polynomial only) - [4,….,10]

- Sample selection - `train_test_split`
  - Train 75% - Test 25% (sklearn defaults).
  - Training - randomly selected 50% Affluent.
  - Model evaluation – adhered to the ‘population’ Acorn proportions: Affluent 40%, Adversity 33%, Comfy 27%



Results

We'll consider two specific models and look at their performance.

SVM Classification Using Gaussian Kernel (RBF)

An example of over-fitting

Model Parameters

- Gaussian Kernel
- $C = 5$
- $\gamma = 0.001$

(parameters found using `GridSearchCV`)

Scoring The Model (Classification Report)

	precision	recall	f1-score	support
Non-Affluent	0.60	0.96	0.74	133
Affluent	0.17	0.01	0.02	88
accuracy			0.58	221
macro avg	0.38	0.49	0.38	221
weighted avg	0.42	0.58	0.45	221

SVM Classification Using Linear Kernel

An example with potential

Model Parameters

- Linear Kernel
- $C = 2$

(parameters found using `GridSearchCV`)

Scoring The Model (Classification Report)

Testing Set

	precision	recall	f1-score	support
Non-Affluent	0.85	0.75	0.80	133
Affluent	0.68	0.80	0.73	88
accuracy			0.77	221
macro avg	0.76	0.77	0.76	221
weighted avg	0.78	0.77	0.77	221

Training Set

	precision	recall	f1-score	support
Non-Affluent	0.93	0.74	0.83	399
Affluent	0.70	0.91	0.79	261
accuracy			0.81	660
macro avg	0.81	0.83	0.81	660
weighted avg	0.84	0.81	0.81	660

ThankYou

Jerry & Michael