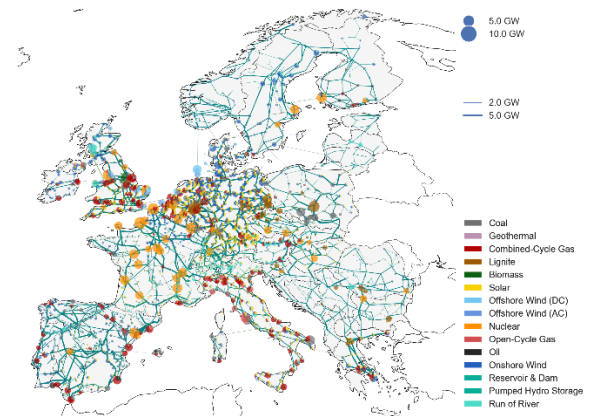


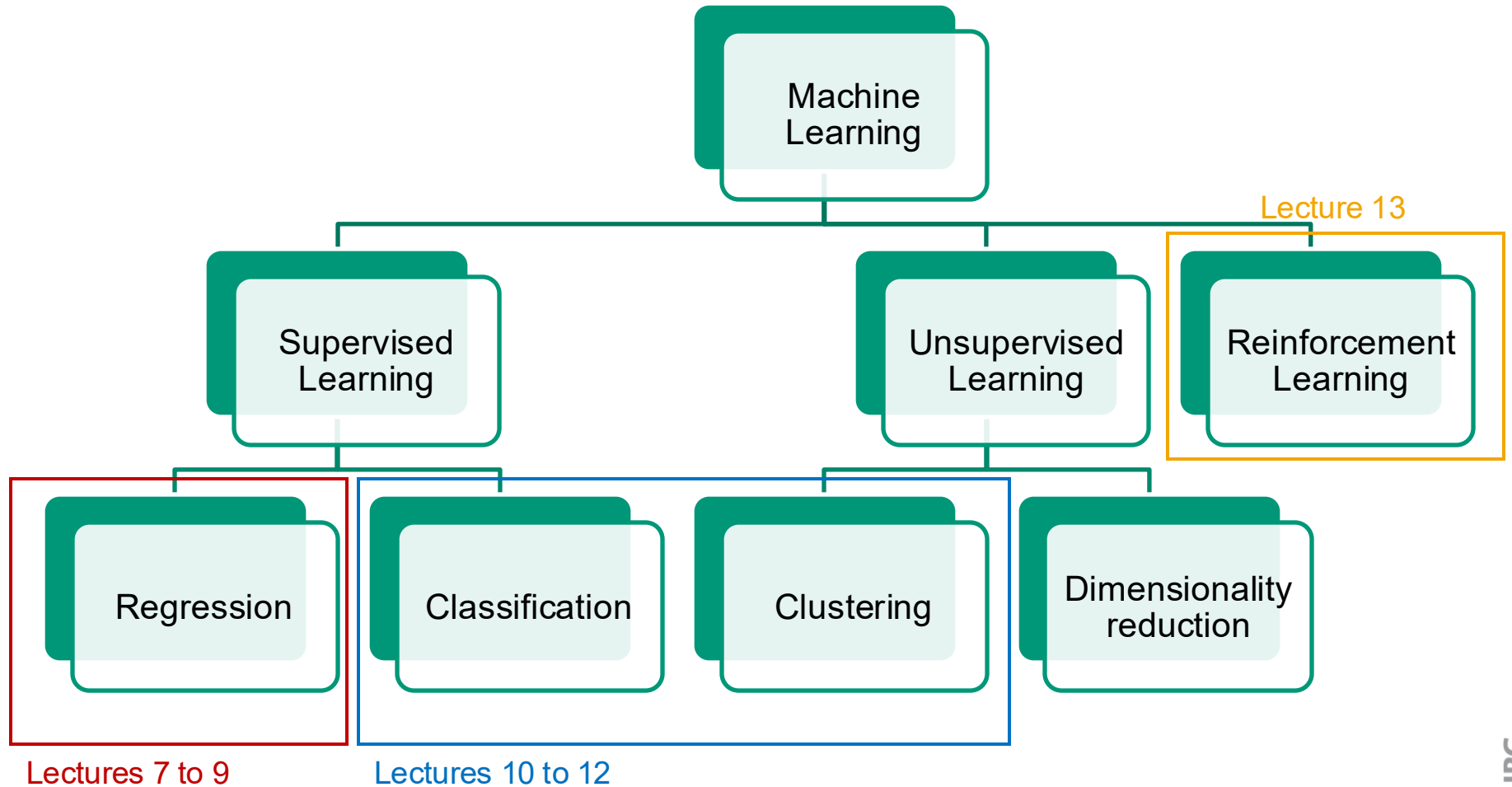
# Energy System Modeling with Python

University of Freiburg (Germany) | Faculty of Engineering  
Department of Sustainable Systems Engineering | INATECH  
**Chair for Control and Integration of Grids**


Tuesday, 1. July 2025



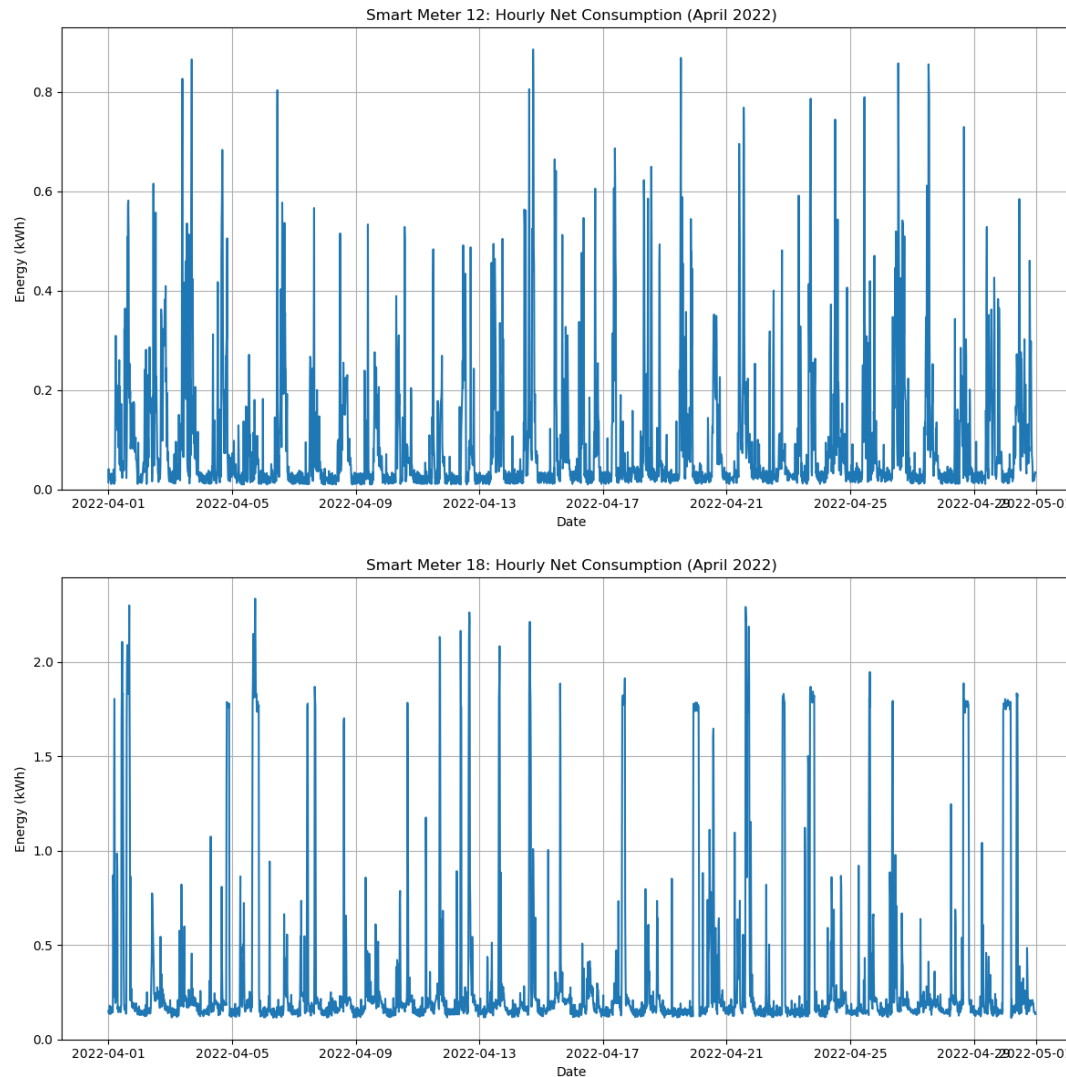
# Branches of ML



# Feature Engineering

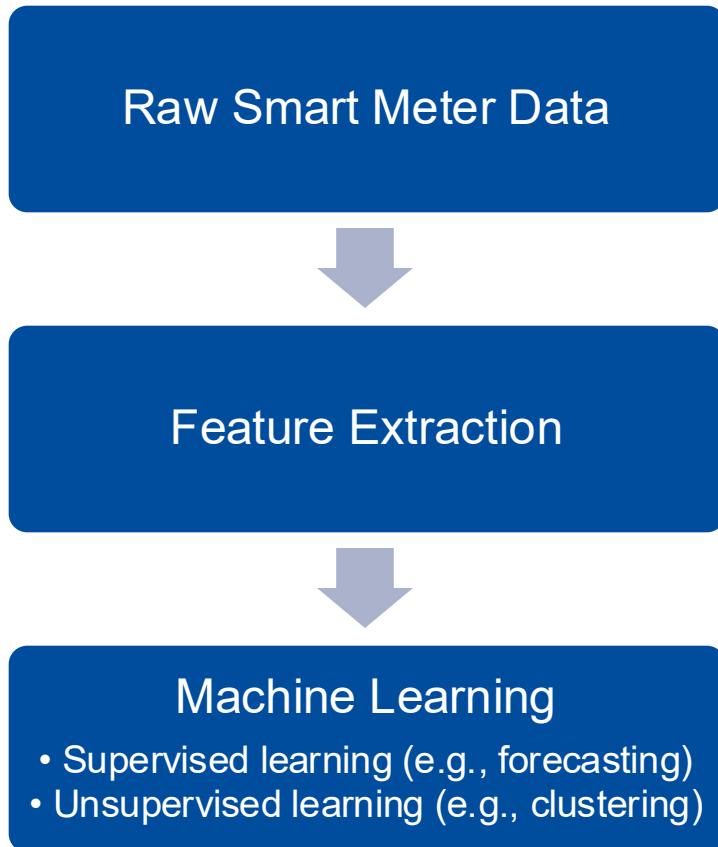


# What Can You Learn from This?



One household has an EV, another one is a baseline household. Which is which?

# From Raw Curves to Insights: Feature Engineering



## Why extract features?

- Reduce dimensionality (from thousands to a few dozen features)
- Make models faster, more robust, and easier to interpret
- Capture key properties: daily shape, variability, peak timing, etc.

## What is a feature?

- A distinctive attribute or aspect of something
- A numeric descriptor summarizing one characteristic of the load curve
- Examples: *night/day ratio*, *midday dip*, *load skewness*, *peak-to-average ratio*

We'll explore two types: **domain-agnostic** and **domain-informed** features

# Domain-Agnostic Features: Descriptors of Time Series

A **domain-agnostic feature** is a characteristic or attribute of data that is **not specific to any industry, task, or application area**, and can be effectively used across multiple domains. These features capture general patterns or structures that are broadly applicable, making them useful in building versatile machine learning models or systems.

## Why use them?

- Work across any kind of time series
- Fast to compute, often highly informative
- Good baseline for clustering

Feature	What it Captures	DER Interpretation Potential
<b>Mean, Std Dev</b>	Central tendency and variability	General activity level, variability
<b>Skewness, Kurtosis</b>	Distribution shape and tail extremity	Detects rare spikes, consumption bias
<b>Autocorrelation (Lag 1)</b>	Short-term dependency / persistence	Operational cycles, device continuity
<b>Autocorrelation (Lag 24)</b>	Daily periodicity / routine	Regular habits, scheduled operations
<b>Entropy</b>	Signal complexity / unpredictability	Behavioral randomness, irregularity
<b>Flat Spots</b>	Periods of constant load (zero variance)	Device idling, standby detection

# Autocorrelation: Temporal Smoothness

---

Autocorrelation measures how similar a time series is to itself at a shifted time (lag). It helps detect:

- Repeated patterns (e.g. daily cycles)
- Memory in the data (persistence)
- Seasonality or trend components

**The autocorrelation at lag  $k$  is:**

$$\rho_k = \frac{\sum_{t=1}^{N-k} (X_t - \mu)(X_{t+k} - \mu)}{\sum_{t=1}^N (X_t - \mu)^2}$$

Where:

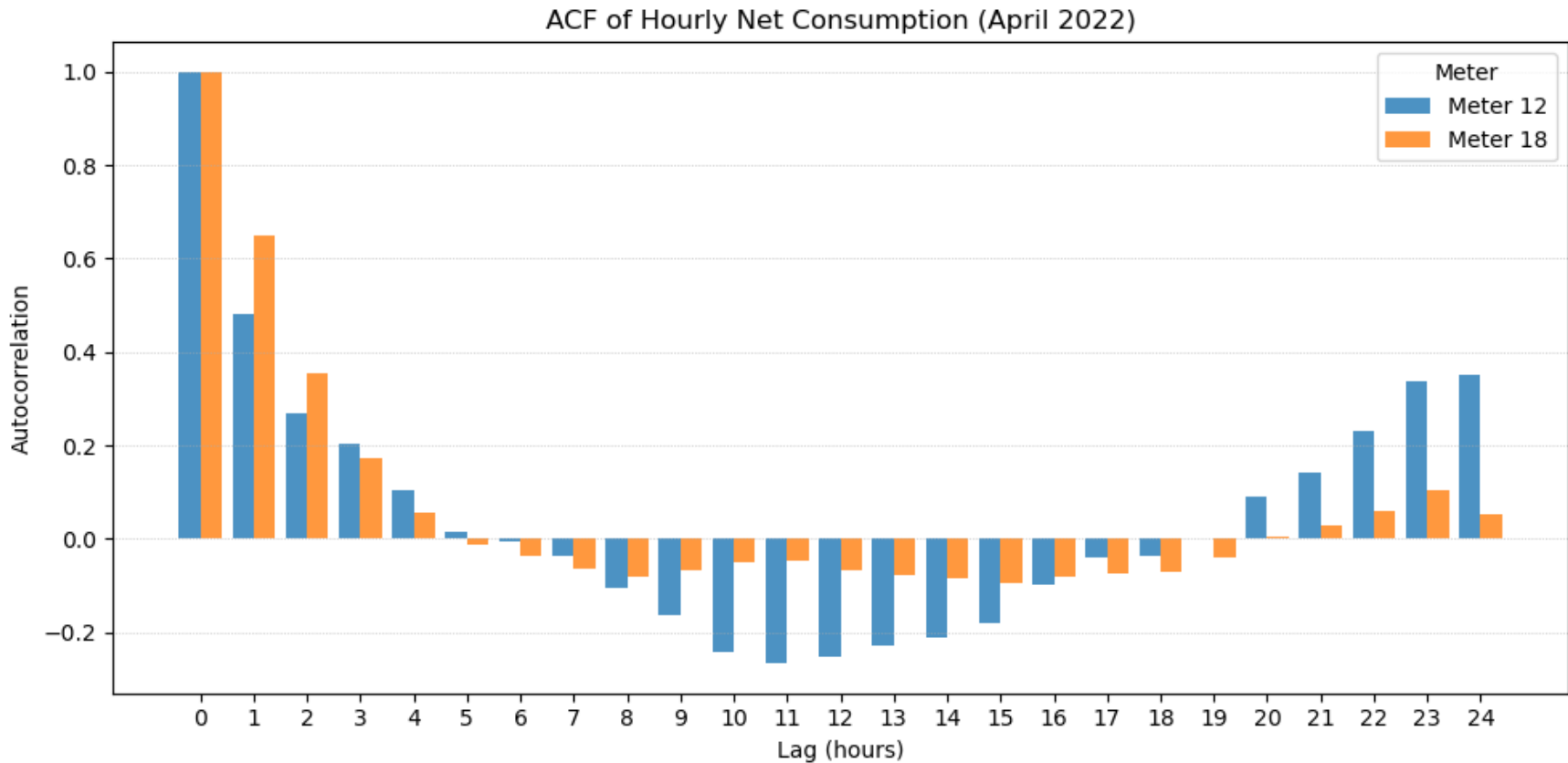
$X_t$ : value of the time series at time  $t$

$\mu$ : mean of the time series

$\sigma^2$ : variance of the time series

$N$ : number of observations

# Autocorrelation: What can you spot?





# Skewness & Kurtosis: Shape of the Distribution

## Skewness (Asymmetry)

$$\text{Skewness} = \frac{1}{N} \sum_{t=1}^N \left( \frac{X_t - \mu}{\sigma} \right)^3$$

where:

$X_t$ : each value in the series

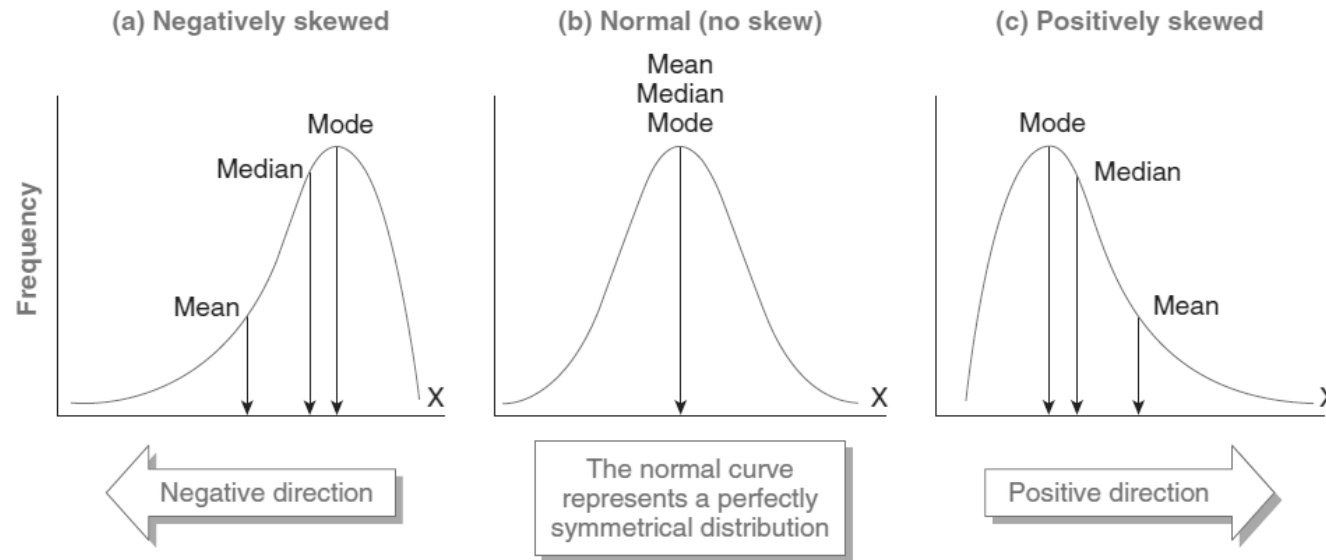
$\mu$ : mean

$\sigma$ : standard deviation

$N$ : number of observations

## Interpretation:

- Positive skew (Skewness > 0) can signal infrequent large loads, like EV charging.
- Negative skew (Skewness < 0) can indicate net export periods from PV generation.



# Skewness & Kurtosis: Shape of the Distribution

## Kurtosis (Tailedness)

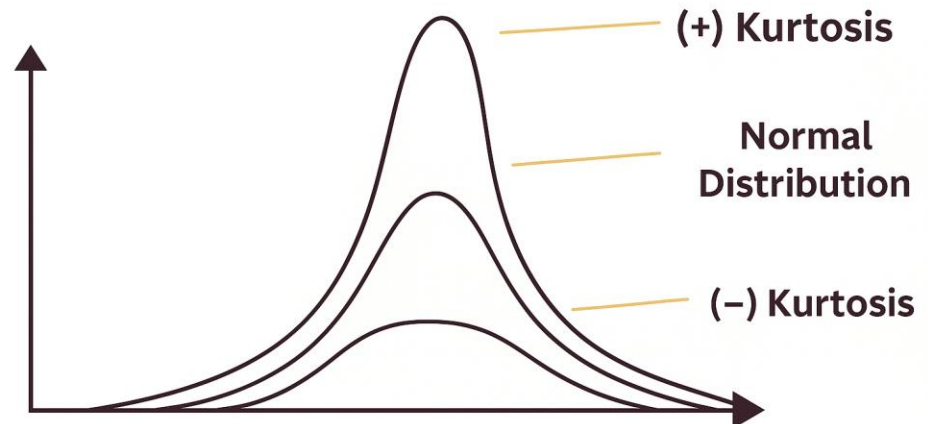
$$\text{Kurtosis} = \frac{1}{N} \sum_{t=1}^N \left( \frac{X_t - \mu}{\sigma} \right)^4$$

Often expressed as **Excess Kurtosis**:

$$\text{Excess Kurtosis} = \text{Kurtosis} - 3$$

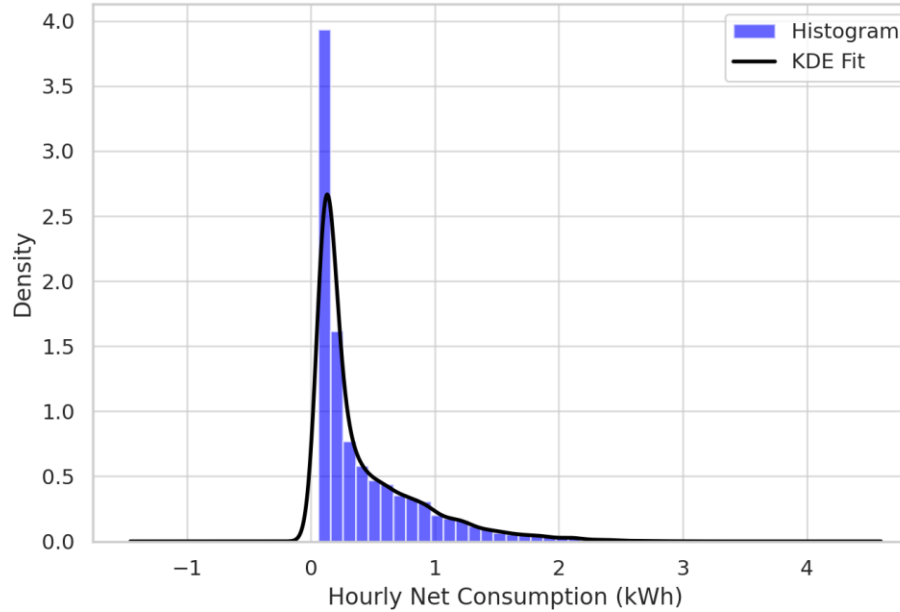
## Interpretation :

- Excess Kurtosis  $\approx 0$ : normal “bell curve” behavior
- Excess Kurtosis  $< 0$ : flatter peak  $\rightarrow$  fewer extreme values
- Excess Kurtosis  $> 0$ : pronounced peak, suggests occasional but extreme consumption or export spikes (e.g., EV charging, high solar feed-in).

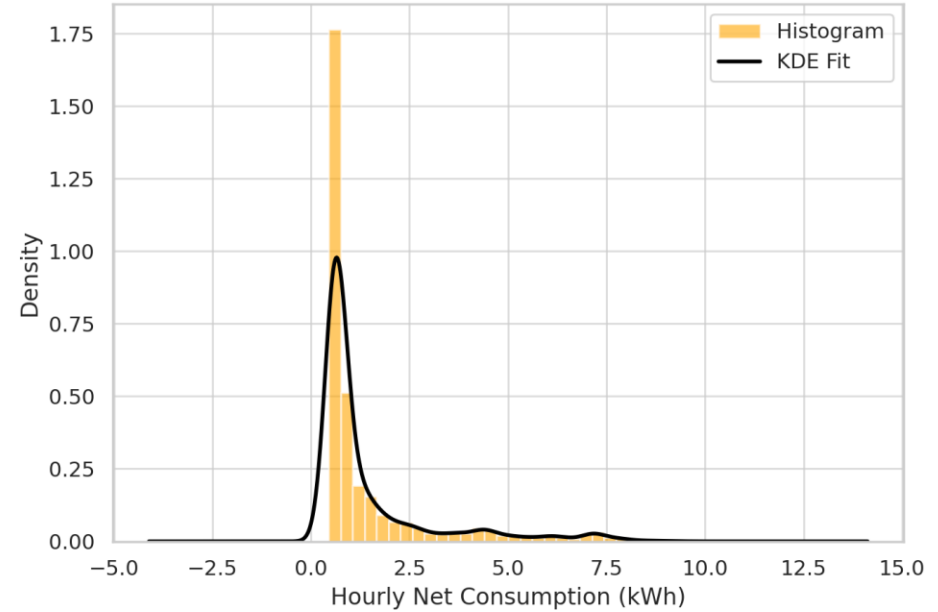


# Skewness & Kurtosis: What can you spot?

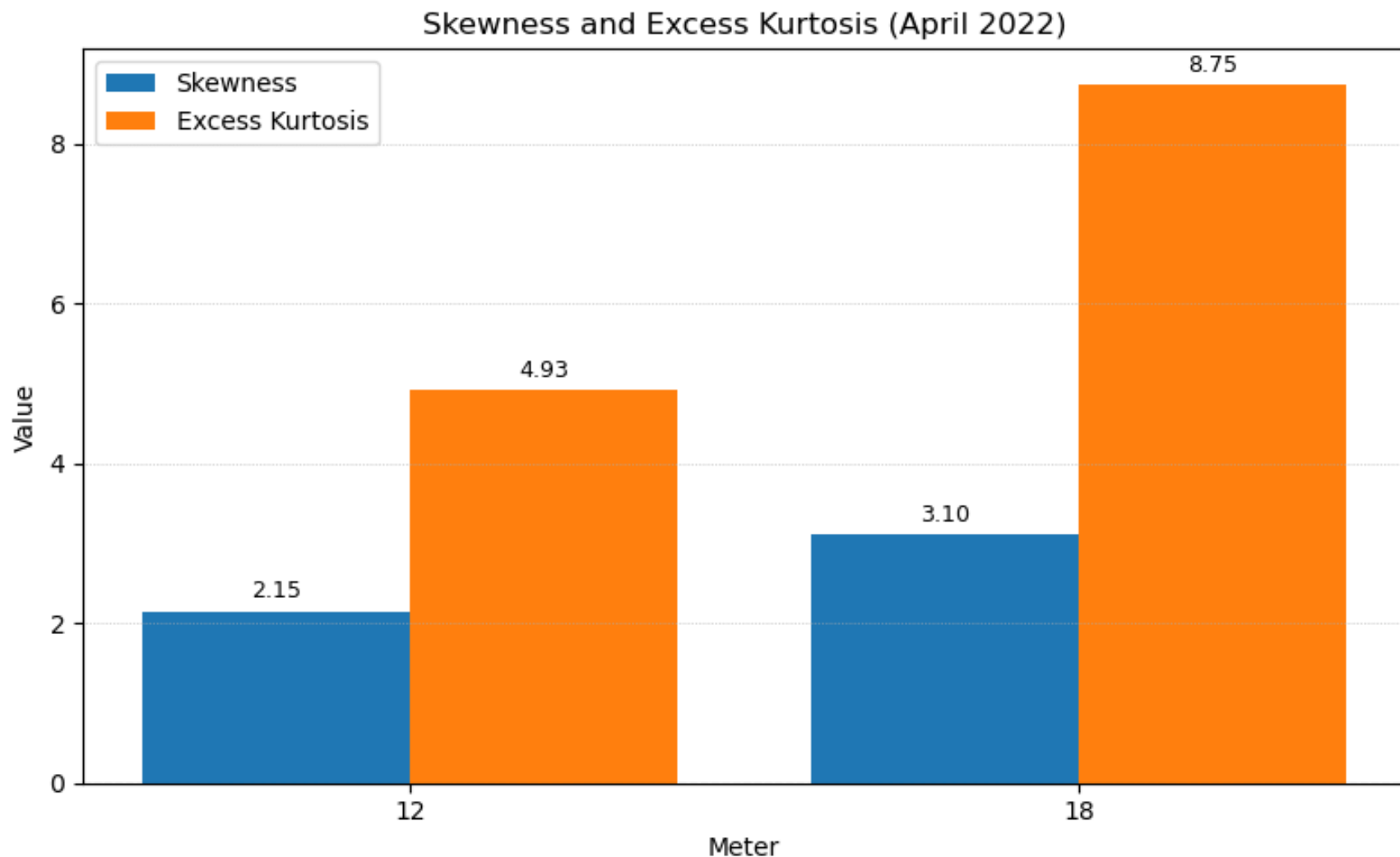
Meter 12 - Hourly Consumption Distribution



Meter 18 - Hourly Consumption Distribution



# Skewness & Kurtosis: What can you spot?



# So, which is which?

Feature	Meter 12 (“Blue”)	Meter 18 (“Orange”)	Interpretation
Lag 1 ACF	0.4815	0.6482	Meter 18’s high lag 1 reflects consecutive EV charging hours. Meter 12’s appliances cycle more frequently.
Lag 24 ACF	0.3499	0.0512	Meter 12 has a strong everyday routine. Meter 18’s charging occurs at varying times, so no daily repeat.
Skewness	+2.15	+3.10	Meter 18’s distribution has fatter right tail (big EV-charging spikes). Meter 12 only has moderate right-tail events.
Excess Kurtosis	+4.93	+8.75	Meter 18 shows extremely heavy tails. Meter 12 has moderate tails from normal appliance peaks.
Lag 2–lag 4 behavior	Decays from 0.27 → 0.11	Decays from 0.36 → 0.05	Meter 18’s multi-hour charging block keeps lag 2 elevated vs. Meter 12’s faster decay.

## Meter 12 is the regular household

- ACF shows a clear daily cycle (lag 24 near +0.35)
- only moderate skew/kurtosis

## Meter 18 is the household with an EV

- It has a very high lag 1 (0.6482) because of multi-hour charging sessions
- almost zero lag 24 (0.0512) because charging times vary day-to-day
- very large skew/kurtosis (3.10, 8.75) due to those extreme charging spikes

# Domain-Informed Features: Using Domain Knowledge

## Definition:

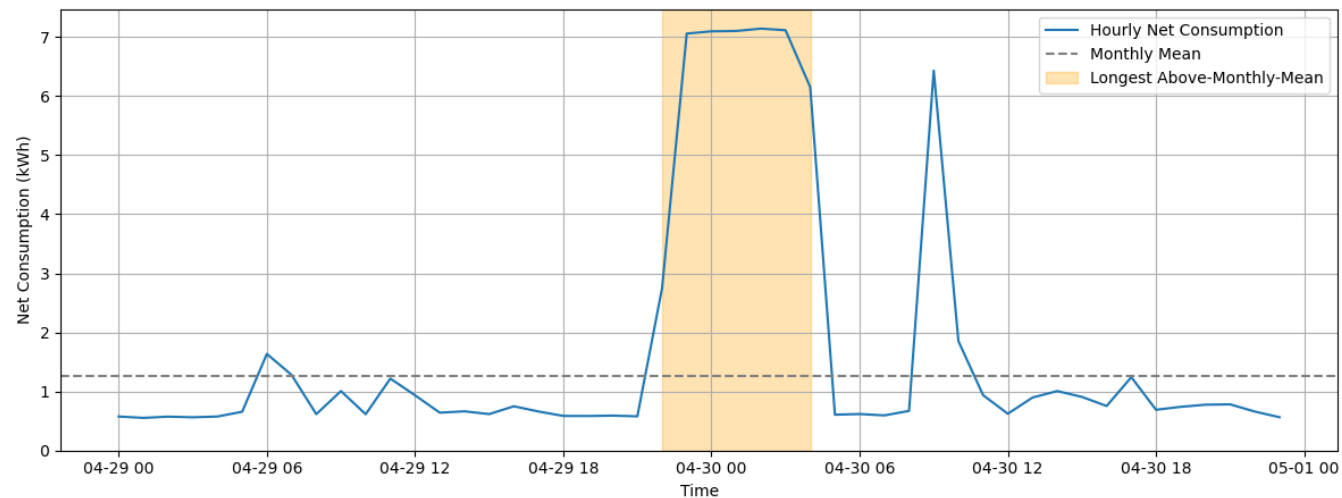
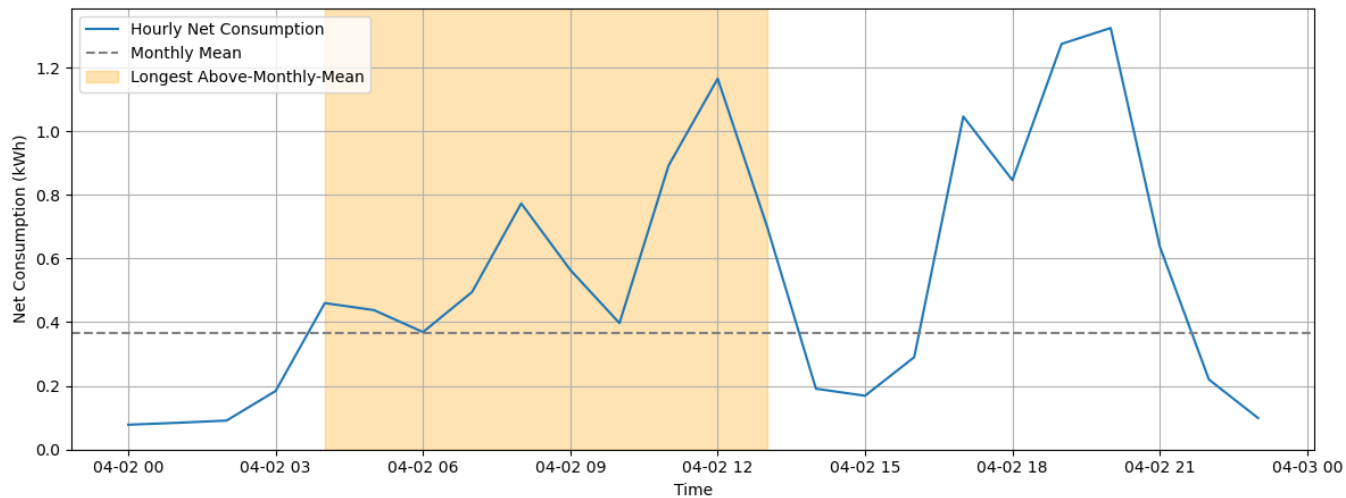
“Features specifically designed to reflect energy use behaviors, patterns of DERs, or daily load structure.”

## Why use them?

- ✓ Capture patterns aligned with human activity (e.g., waking hours, night/day load balance)
- ✓ Reveal DER-specific markers (e.g., midday dip from PV, spikes from EV)
- ✓ More interpretable for grid planners and energy analysts

Feature	What it Captures	DER Interpretation Potential
<b>Longest Period Above Mean</b>	Sustained high consumption	Heat pump / EV
<b>Longest Successive Increase</b>	Structured ramp-ups	EV charging / manual load
<b>Midday Dip</b>	Solar PV export impact	PV detection
<b>Night/Day Ratio</b>	Load shifting between periods	Behavioral segmentation

# Longest Period Above Mean: Sustained High Load



# Feature Engineering: Summary

## Domain-Agnostic Features

Feature	What it Captures	DER Interpretation Potential
<b>Mean, Std Dev</b>	Central tendency and variability	General activity level, variability
<b>Skewness, Kurtosis</b>	Distribution shape and tail extremity	Detects rare spikes, consumption bias
<b>Autocorrelation (Lag 1)</b>	Short-term dependency / persistence	Operational cycles, device continuity
<b>Autocorrelation (Lag 24)</b>	Daily periodicity / routine	Regular habits, scheduled operations
<b>Entropy</b>	Signal complexity / unpredictability	Behavioral randomness, irregularity
<b>Flat Spots</b>	Periods of constant load (zero variance)	Device idling, standby detection

## Domain-Informed Features

Feature	What it Captures	DER Interpretation Potential
<b>Longest Period Above Mean</b>	Sustained high consumption	Heat pump / EV
<b>Longest Successive Increase</b>	Structured ramp-ups	EV charging / manual load
<b>Midday Dip</b>	Solar PV export impact	PV detection
<b>Night/Day Ratio</b>	Load shifting between periods	Behavioral segmentation

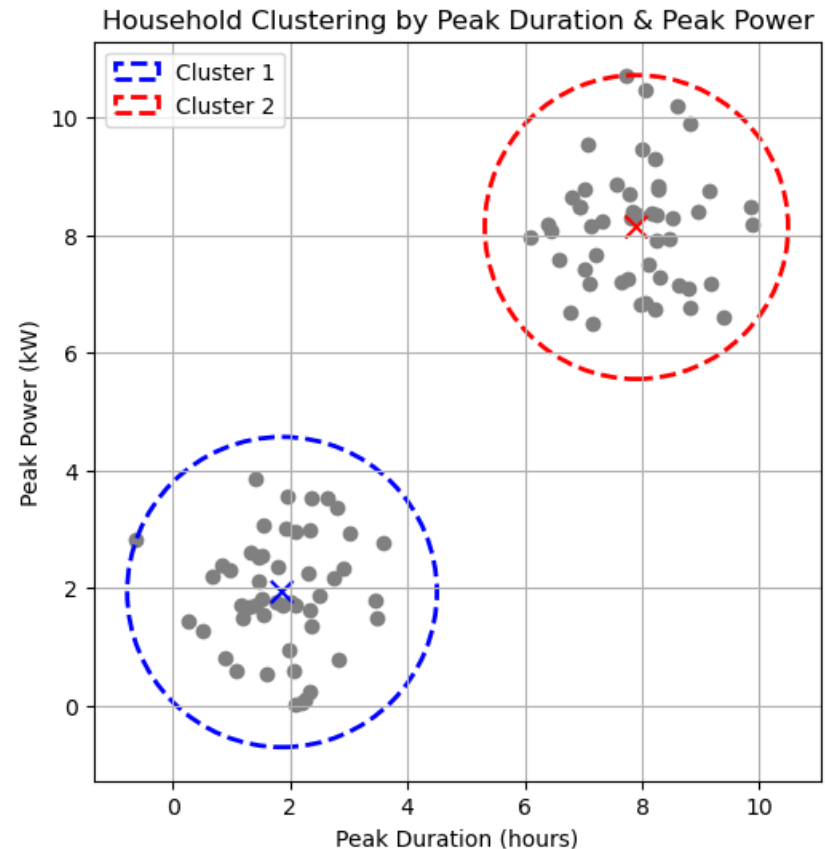




# Clustering

# Why Clustering? And What is Clustering?

- Manual comparison helped distinguish a few known profiles (e.g., EV vs baseline)
- But this doesn't scale to 1,000s of households
- We want to **group households automatically** based on extracted features
- We don't always know how many types (clusters) exist — sometimes it's known, sometimes not



**Definition:** The clustering consists in grouping a set of objects so that members of the same group (called cluster) are more similar.

# K-Means: Grouping by Proximity in Feature Space

## Core idea:

Given data  $\{x_i\}_{i=1}^n$ , where  $x_i \in R^d$ , partition into  $k$  clusters  $C_1, \dots, C_k$  to minimize the **within-cluster sum of squares**:

$$\min_{C_1, \dots, C_k} \sum_{j=1}^k \sum_{x_i \in C_j} |x_i - \mu_j|^2$$

Where  $\mu_j$  is the **centroid** of cluster  $C_j$ :

$$\mu_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$$

## Algorithm:

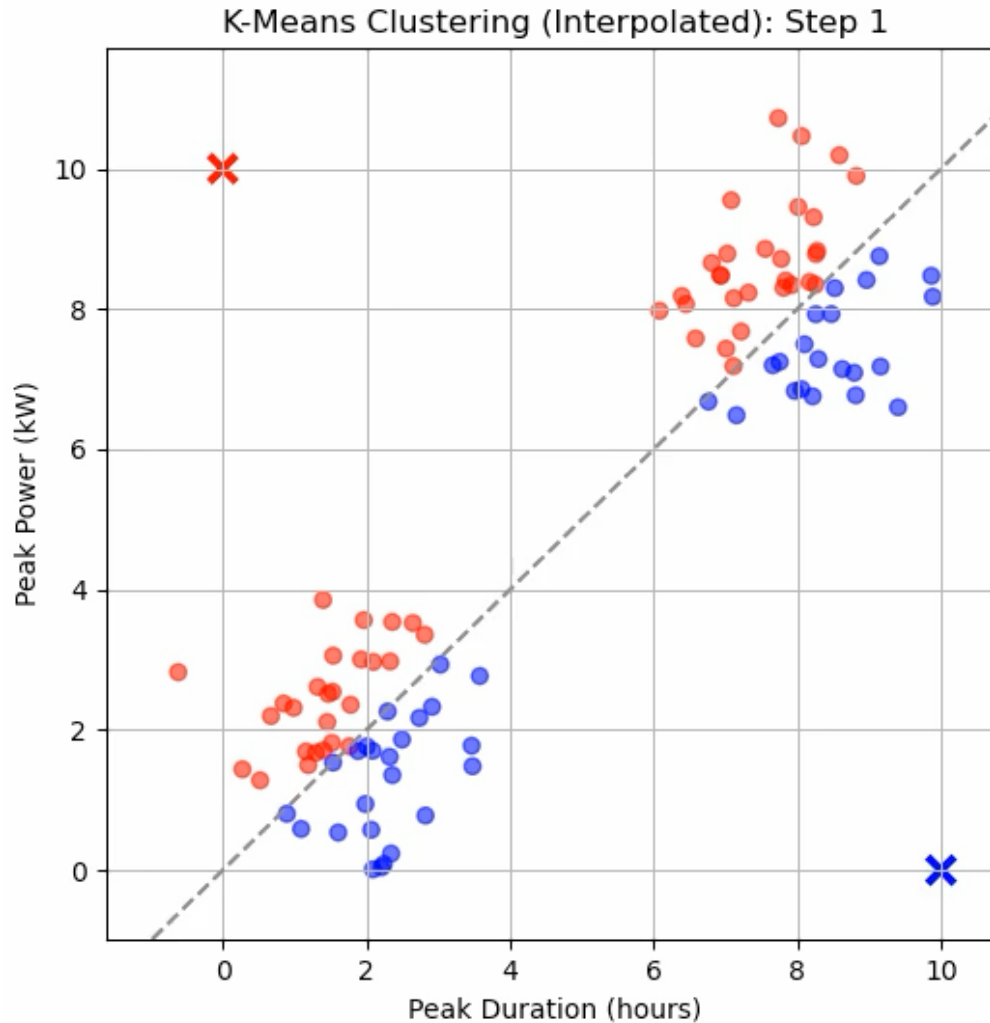
1. Initialize  $\mu_1, \dots, \mu_k$  randomly
2. Repeat until convergence:
  - Assignment step:

$$c(i) = \arg \min_j |x_i - \mu_j|^2$$

- Update step:

$$\mu_j = \frac{\sum_{i=1}^n \mathbb{1}(c(i) = j) \cdot x_i}{\sum_{i=1}^n \mathbb{1}(c(i) = j)}$$

# K-Means: Grouping by Proximity in Feature Space



# Ward's Method: Bottom-Up Clustering with Variance Minimization

## Core idea:

Agglomerative clustering builds a **hierarchical tree** by **iteratively merging** the two closest clusters.

With **Ward linkage**, the "closeness" is defined by how much the total **within-cluster variance** would increase if two clusters are merged.

At each step, choose the pair  $(A, B)$  of clusters to merge that minimizes:

$$\Delta E(A, B) = \frac{|A| \cdot |B|}{|A| + |B|} \cdot |\mu_A - \mu_B|^2$$

## Where:

$|A|, |B|$  are the sizes of clusters A and B

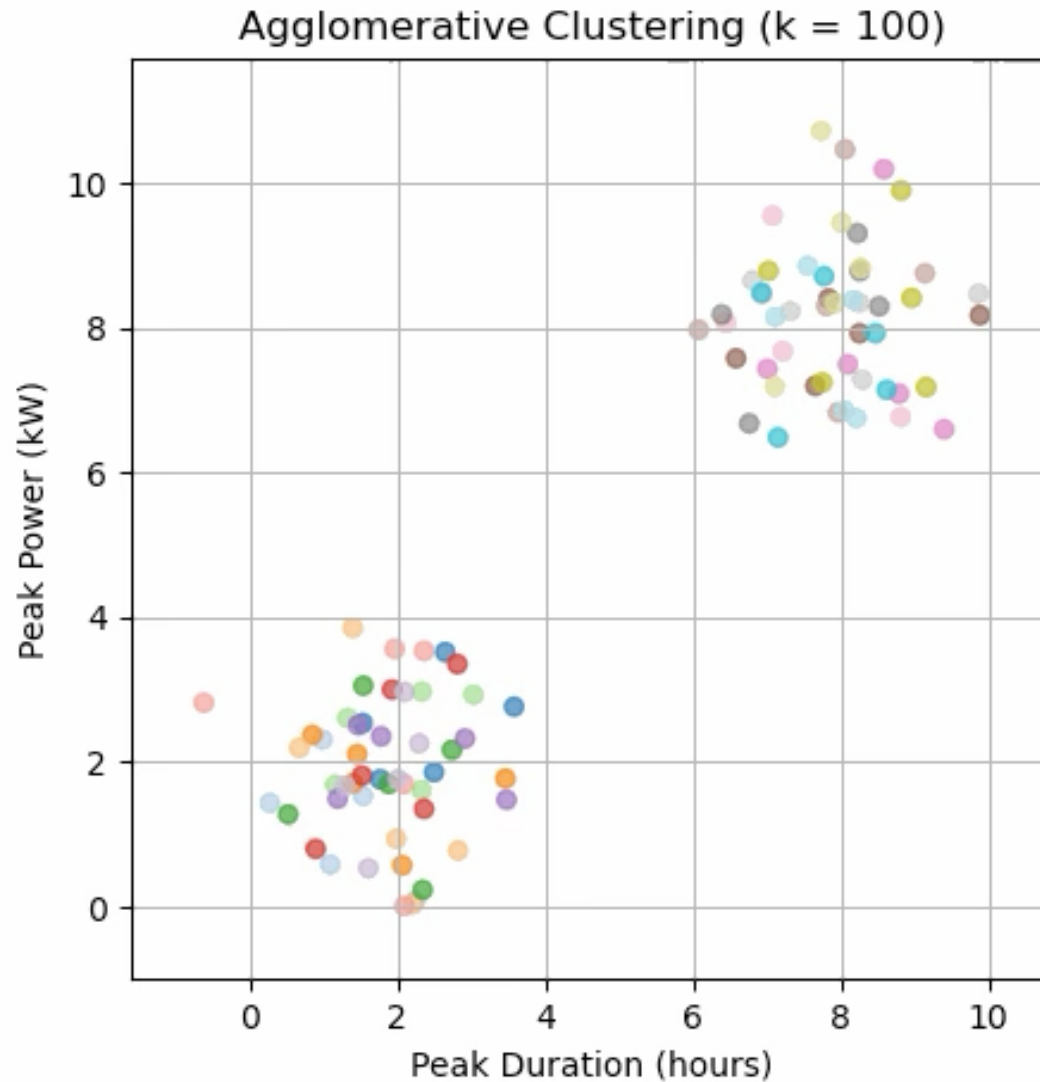
$\mu_A, \mu_B$  are their centroids

$\Delta E$  is the increase in total within-cluster sum of squares

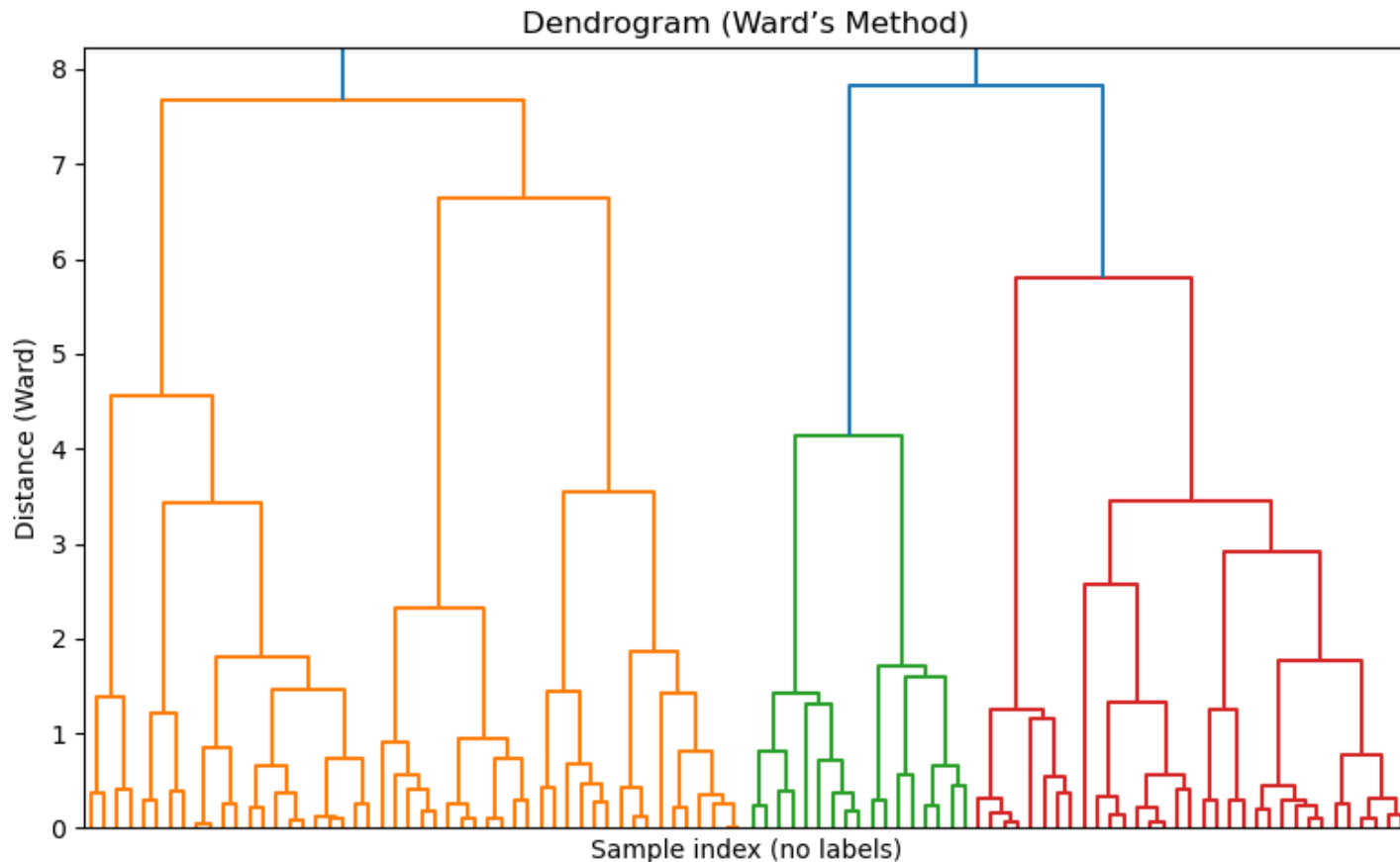
## Algorithm:

1. Start with each data point as its own cluster.
2. Compute all pairwise cluster distances.
3. Merge the pair with the **smallest increase in variance**.
4. Repeat until a single cluster remains or desired number of clusters is reached.

# Ward's Method: Bottom-Up Clustering with Variance Minimization



# Ward's Method: Dendrogram



A **dendrogram** is a tree-like diagram that shows how data points or clusters are merged (or split) during **hierarchical clustering**.

It's a **visual map of the clustering process** — showing how each point starts in its own cluster and how clusters get merged step by step until only one cluster remains.

# Choosing a Clustering Method: Trade-Offs

Aspect	K-Means	Agglomerative (Ward)
<b>Cluster shape assumption</b>	Spherical, equal-sized clusters	Any shape, hierarchical structure
<b>Input requirement</b>	Must specify number of clusters	No need to specify number of clusters upfront (optional cutoff later)
<b>Distance metric</b>	Euclidean (squared L2 norm)	Increase in within-cluster variance (Ward linkage)
<b>Scalability</b>	Fast on large datasets	Slower and memory-heavy
<b>Output structure</b>	Flat clusters only	Hierarchical tree (dendrogram)
<b>Determinism</b>	Random init → results may vary (use multiple runs)	Deterministic
<b>Interpretability</b>	Easy to visualize centroids	Reveals structure at multiple scales
<b>Use case fit (for DER)</b>	Clear household types known (e.g. EV, PV, baseline)	Unknown number/types of behaviors; exploratory profiling

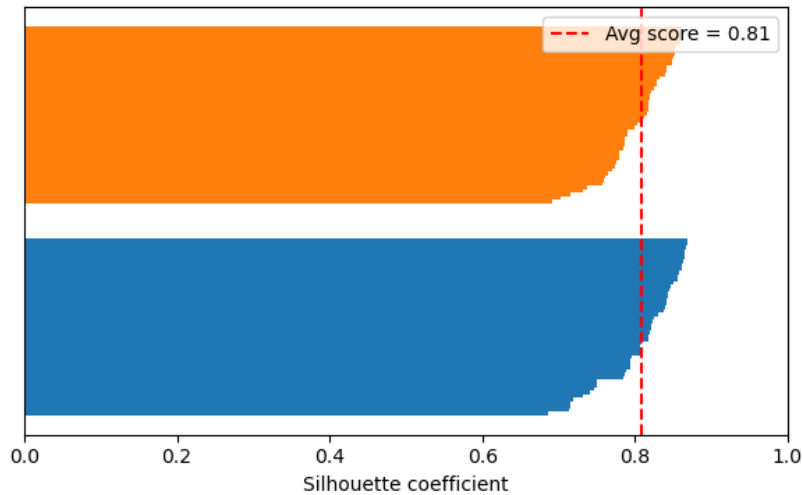


# How Do We Know If Clustering Worked?

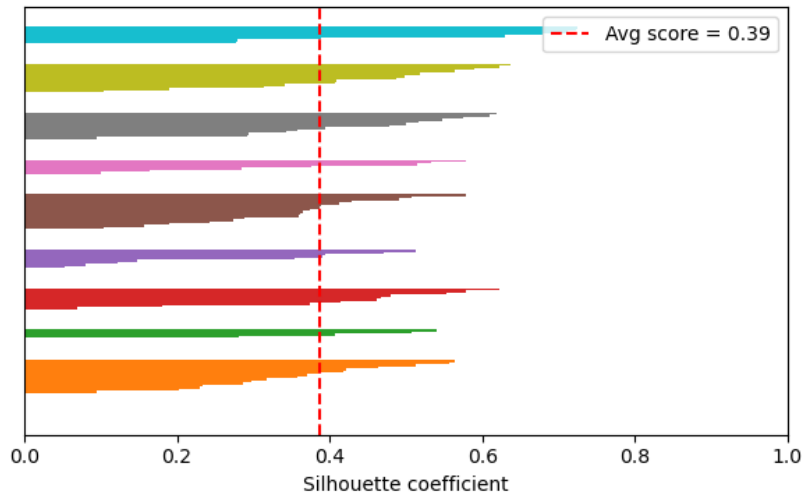
Aspect	Definition / Equation	Purpose	Key Notes
<b>Silhouette Score</b>	$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$ $a_i$ : intra-cluster distance $b_i$ : nearest-cluster distance	Measures cluster compactness and separation	Values close to 1 = good clustering Near zero = misclassified
<b>Elbow Method</b>	Plot inertia score vs. number of clusters  Inertia = $\sum_{i=1}^k \sum_{x \in C_i}  x - \mu_i ^2$ : sum of squared distances between each point and the centroid of its assigned cluster	Identify optimal number of clusters	Look for "elbow" — point where additional clusters give diminishing returns
<b>t-SNE</b>	Nonlinear dimensionality reduction Preserves local neighborhoods; not distances	Visualize clusters in 2D	For <b>visualization only</b> , not cluster assignment Layout may change on re-run

# How Do We Know If Clustering Worked?

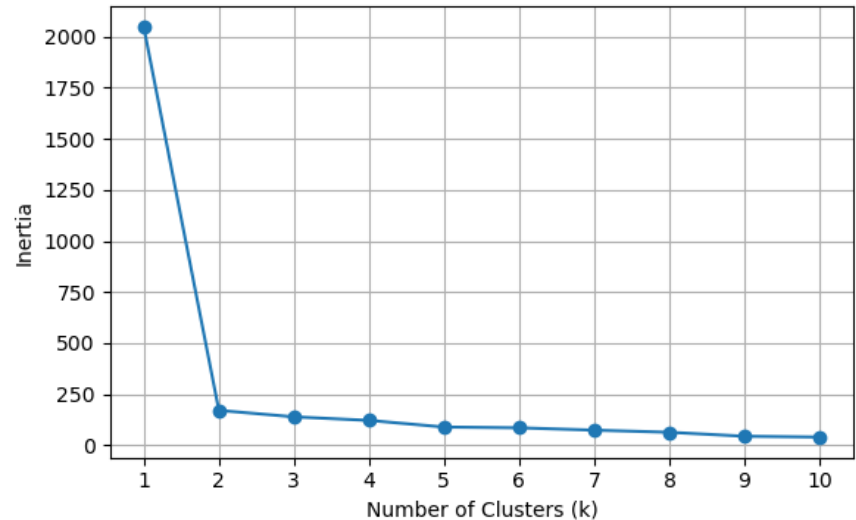
Silhouette Plot for k=2



Silhouette Plot for k=10



Elbow Method for Optimal k



# Coffee Break

---



# Time to put everything into code

**START  
CODING  
SESSION**



# What You'll Do in Code Today

---

## Goal:

Use smart meter data to extract meaningful features, apply clustering, and explore how different types of households group together based on their electricity usage.

## Main Steps:

1. **Load and explore** the dataset of household electricity profiles.
2. **Extract features** that summarize daily and weekly consumption patterns.
3. **Standardize** the features to prepare them for clustering.
4. **Visualize** the data structure using dimensionality reduction.
5. **Apply clustering** using two different algorithms and compare the results.
6. **Interpret** the discovered clusters and connect them to possible DER types.

# Takeaways

---

- ✓ Raw smart meter data is high-dimensional and hard to interpret directly — we need feature engineering to extract meaningful patterns.
- ✓ Domain-agnostic features offer general, statistically grounded insights; useful for broad structure detection.
- ✓ Domain-informed features capture behaviorally or technically relevant characteristics (e.g., PV dips, EV ramps).
- ✓ Clustering methods like K-Means and Ward help group households with similar load behavior — even without labeled data.
- ✓ Evaluation tools such as silhouette scores and t-SNE plots help assess and interpret clustering outcomes.

## Further Questions to Think About

- Which features are most relevant for detecting specific DERs like heat pumps or batteries?
- How sensitive are clustering results to the feature set or scaling method used?
- How might time-of-day or seasonal effects influence feature extraction or clustering quality?