Funded by
the European Union

Horizon Europe
EUROPEAN COMMISSION
European Climate, Infrastructure and Environment Executive Agency (CINEA)
Grant agreement no. 101056765

ev4eu

Electric Vehicles Management for carbon neutrality in Europe

Deliverable D3.3
EVs use Clustering results report

Document Details

Due date	30-11-2023
Actual delivery date	05-01-2024
Lead Contractor	Public Power Corporation (PPC)
Version	1.0
Prepared by	Alexios Lekidis (PPC), Marcelo Forte (INESC ID), Cindy P. Cuzman (INESC ID), George Papadakis (PPC), Nikos Iliopoulos (PPC), Angelos Georgakis (PPC)
Reviewed by	Matej Zajc (UL), Hugo Morais (INESC ID)
Dissemination Level	Public

Project Contractual Details

Project Title	Electric Vehicles Management for carbon neutrality in Europe
Project Acronym	EV4EU
Grant Agreement No.	101056765
Project Start Date	01-06-2022
Project End Date	30-11-2025
Duration	42 months

Document History

Version	Date	Contributor(s)	Description
0.1	21/09/2023	Alexios Lekidis (PPC)	Table of contents
0.2	31/10/2023	Marcelo Forte (INESC ID), Cindy P. Cuzman (INESC ID)	Background, Clustering algorithms (K-means, GMM and Hierarchical), Preliminary Data Analysis, Methodology, First Results
0.3	30/11/2023	Nikos Iliopoulos (PPC)	Refinement of existing content
0.4	15/11/2023	Angelos Georgakis (PPC)	Graph clustering experiments
1.0	21/12/2023	George Papadakis (PPC)	Text finalization
1.1	29/12/2023	Matej Zajc (UL), George Papadakis (PPC)	Refinements based on internal review
1.2	05/01/2024	Ana Rita Nunes (INESC ID), Hugo Morais (INESC ID), George Papadakis (PPC), Nikos Iliopoulos (PPC)	Refinements based on internal review

Disclaimer

This document has been produced in the context of the EV4EU[1] project. Views and opinions expressed in this document are however those of the authors only and do not necessarily reflect those of the European Union or the European Climate, Infrastructure and Environment Executive Agency (CINEA). Neither the European Union nor the grating authority can be held responsible for them.

Acknowledgment

This document is a deliverable of EV4EU project. EV4EU has received funding from the European Union´s Horizon Europe programme under grant agreement no. 101056765.

Funded by
the European Union

[1] https://ev4eu.eu/

Executive Summary

The present deliverable 3.3 “EVs use Clustering results report” aims to provide a set of services adapted to different needs and user behaviour based on clustering EV profiles. To this end, it was applied an AI methodology that consists of the following steps:

1. Data cleaning and pre-processing removes noisy records with missing or outlier values.
2. Feature engineering defines new comprehensive features and reduces the features to the most essential, non-correlated ones.
3. Clustering applies an unsupervised learning algorithm to generate the final sets of records with common patterns.

Domain-specific approaches are applied for the first two steps, while the third one involves 10 state-of-the-art clustering algorithms that cover all main types of approaches, from representative-based to graph-based and hierarchical ones.

This methodology is applied to a dataset with real data from PPC's nationwide network fo EV chargers. Through a thorough experimental analysis, all clustering methods were fine-tuned with grid search, with K-Means achieving the best performance among all algorithms for K=10. The resulting profiles capture quite diverse patterns of behavior, providing useful insights that can be used to define demand-response programs in Task 4.5 and optimal aggregation by virtual power plants in Task 4.3.

The deliverable D3.3 has been prepared and edited by PPC with the support of INESC ID.

Table of Contents

List of Figures

List of Tables

Acronyms

AC	Alternate Current
CPO	Charge Point Operator
CS	Charging Station
CSMS	Charging Station Management System
DC	Direct Current
DSO	Distribution System Operator
EM	Expectation-Maximization
EV	Electric Vehicle
EVSE	Electric Vehicle Supply Equipment
GMM	Gaussian Mixture Model
SSE	Sum of Squared Errors
SR	Sequential Rippling
V2X	Vehicle-to-Everything

1 Introduction

1.1 Scope and Objectives

Task 3.3 “EVs Clustering in Cities Management” aims to outline and describe in detail the usage of AI techniques to cluster EV users according to charging needs, places of charging, and parking time. Its scope includes all the activities related to clustering such as algorithms, data collecting and processing. Concretely, the objectives of this deliverable are as follows:

- AI techniques and algorithms description and application for users clustering.
- Data processing from >400 public charging stations and >3,000 charging network users.
- Presentation of the results of clustering.

Deliverable 3.3 “EVs use Clustering results report” is the main output of Task 3.3 and with its submission the task is completed.

1.2 Structure

The current document is divided into six sections. Section 1 provides an overview of this deliverable, while Section 2 provides an overview of the related works in the field. In Section 3, we briefly describe the clustering algorithms that were used in the analysis of the EV charging data. Section 4 reports the pre-processing of the real data extracted from PPC’s nationwide network of EV chargers. Section 5 elaborates on the methodology that was applied on the clean data. Finally, Section 6 provides the results and conclusions on this deliverable.

1.3 Relationship with other deliverables

Deliverable D3.3 presents the detailed description and application of AI techniques and algorithms for users clustering. The results of this deliverable will be used to estimate the needs of different clusters of users and, consequently, define the energy needs in each of the clusters and each parking lot for task T3.6. Also, the results will be used to define Demand Response programs in T4.5 and optimal aggregation by Virtual Power Plants in T4.3. Hence, this deliverable is relevant with the deliverables concluding the work in these tasks, namely **D3.6 “High-Level Design of V2X Management Strategies Coordination”, D4.5 “Demand Response Programs Design for EVs”, D4.3 “Integration of V2X in Charging Point Operators and Virtual Power Plants Aggregation”**.

2 Background

2.1 EV Charging Process

In the context of EV charging, the terms Electric Vehicle Supply Equipment (**EVSE**) and Charging Station (**CS**) are often used interchangeably. Yet, there is a slight difference between the two terms. While commonly referred to as a "charger" or a "charging point", an EVSE is technically the equipment that provides electricity to an EV. On the other hand, a CS is a physical object that includes one or more EVSEs sharing a common user identification interface (similar to a gas pump with multiple refuelling hoses for classical cars, with internal combustion engines). A site with one or more CSs and the associated parking lots is known as a Charging Pool (**CP**) and is operated and managed by a Charging Point Operator (**CPO**).

Regarding the charging process, conductive EV charging can be divided into three categories:

1. Level 1, which involves a slow charging process, using a regular 120-volt wall plug, found in all houses and garages,
2. Level 2, which requires a dedicated 240 volt charger, but it's 15 times faster than Level 1, and
3. Level 3, mostly known as DC fast charging, which uses 480+ volts, found in public places.

An EV receives Alternate Current (**AC**) from Level 1 and Level 2 chargers, which is then converted to Direct Current (**DC**) internally by the EV (through a slow process), because EV batteries exclusively support DC power. In contrast, no conversion is necessary when using a DC fast EVSE. Level 1 and Level 2 chargers typically use Type 1 connectors in America (SAE J1772), while for European and Asian vehicles, Type 2 connectors are standard. Dimitriadou et al. [1] present an overview of the current status of the infrastructure utilized for the realization of both conductive and wireless charging of an EV battery, presenting a detailed exposition of the respective standards and charging levels, as well as future challenges and opportunities.

2.2 EV Charging Profiles

Working with a large dataset from metropolitan areas of the Netherlands, Helmus et al. [2] carried out a two-step, bottom-up data clustering approach that first employs Gaussian Mixture Model (**GMM**) to cluster charging sessions and then portfolios of charging sessions per user using K-Medoids (an approach similar to K-means clustering). The study considers starting time, connection duration, the distance between two sessions, and hours between sessions as features. From the first step, thirteen clusters were found: 7 types of daytime and 6 types of nighttime charging sessions. The second step resulted in nine distinct clusters: 3 clusters contained daytime users, 3 nighttime, and the other 3 featured unusual users.

Martz et al. [3] claim that they used the most extensive (private) dataset on charging patterns from an EV perspective known in the literature, containing approximately 21 000 BMW i3 Battery EVs and about 2.6 million charging processes during one year (2019). The authors performed GMM clustering on the EV charging behavior, utilizing plug-in time and duration as features, and discovered seven distinct clusters: 3 overnight and 4 daytime. The authors conducted a second analysis with K-means clustering to identify switching EV users between clusters. They also made known the flexibility potential of the EV charging processes, concluding that there was a huge potential: on average, the temporal flexibility was 8 hours. The methods are well described, and the decisions are thoroughly justified, leading to outstanding illustration and understanding of the characteristics of the clusters found, turning this analysis into one of the most complete in the literature.

Shahriar and Al-Ali [4] performed cluster analysis with K-means, Hierarchical clustering, and GMM to identify similar groups of charging behavior, based on EV arrival and departure times, on real public EV charging activity during the COVID-19 pandemic. K-means produced the best results, followed by Hierarchical clustering. The authors only discovered three clusters corresponding to the knee of the elbow method curve. The study's drawbacks include only employing a single method for establishing the appropriate number of clusters and selecting only two features to group the data: arrival and departure times, resulting in generic results.

The K-means technique was also employed by Shen et al. [5] to identify charging behavior clusters. The authors supervised the clustering results and adjusted them to achieve the best possible outcomes, something fundamental when data is sparse and/or irregular. To obtain typical user behavior, the data was grouped by user, leading to the average charging time, the standard deviation of charging time, and the standard deviation of connection time as the basis for the clustering. Three clusters were discovered. Two groups were identified as stable and predictable users, but the third cluster comprised unexpected users.

Similarly, Xiong et al. [6] attempted to find EV user behavior by organizing the data by the user. Thus, each user was represented by the following features: average arrival time, average departure time, standard deviation of arrival time, standard deviation of departure time as well as the Pearson correlation coefficient, between stay duration and energy consumption. With this data, they performed clustering with K-means, obtaining four profiles.

Van Kriekinge et al. [7] proposed a methodology to simulate the charging demand for different types of drivers. Typical EV driver profiles with similar charging habits are needed to accomplish this goal. To obtain user behavior profiles, all charging sessions from a private dataset were replaced by one specific theoretical charging session per EV driver represented by the average value of the plug-in times, parking times, and charged energy, yielding the mean behavior per driver. The clustering proposed in this study works in two stages: cluster the average characteristics per EV user and then analyze the frequency of charging, always with the K-means algorithm. The results indicated five clusters, with big differences in behavior between the EV drivers. In addition, the Kernel Density Estimation (KDE) process allows capturing the details of each cluster, helping in the final simulation stage, which demonstrated a strong impact on power and energy demand when adding new EV users to the population.

Gerossier et al. [8] employed hierarchical clustering to identify four groups of EV charging behavior. The authors received data in time-series format, which they processed to extract individual sessions categorized by start-up time (initial plug-in time) and duration of the charging process, following a method well-described and presented in the study. Most customers belonged to the first group, where charging was typically performed during the evening and morning.

The above works are summarized below, in Table 1, with respect to their main characteristics.

Table 1. Summary of the most relevant works on clustering EV charging sessions.

Study	Brief summary	Clustering method	Dataset	Conclusions
Helmus et al. [2] Amsterdam, Netherlands 2020	Provides a realistic analysis of charging behavior and EV user types based on clustering, differing from the typical literature that is frequently oversimplified.	GMM for clustering and Partition Around Medoids to find portfolios of charging sessions per user	5.82 million charging transactions (January 2017-March 2019) from the Dutch metropolitan area	13 clusters were found: 7 types of daytime charging sessions (4 short, 3 medium duration) and 6 types of overnight charging sessions.
Martz et al. [3] Germany 2022	Investigates the possibility of identifying different clusters of EV charging processes, validating the results against synthetic load profiles and the original data.	GMM and K-means	2.6 million private charging processes of 21 000 BMW's i3 model from 2019 in Germany	High number of charging opportunities during day, as well as user exchange between charging clusters, to reduce localized energy demand. Found 7 clusters.
Shahriar and Al-Ali [4] UAE 2022	Investigates the impacts of COVID-19 on EV charging behavior by analyzing the charging activity during the pandemic.	K-means, Hierarchical clustering, and GMM	ACN dataset, from Caltech University Campus	Identified 3 groups of charging behavior. The best clustering was obtained using K-means followed by Hierarchical clustering.
Shen et al. [5] USA & Canada 2020	To manage (dis)charging behavior of EVs in the smart grid, proposes a communication network for analysis and prediction of user behavior.	HITL-based K-means clustering and K-NN algorithm for prediction	ACN dataset, from Caltech University Campus	Identified 2 clusters of stable, predictable users, but the third cluster was found to be unexpected users.

Xiong et al. [6] Los Angeles, USA 2018	Proposes an EV user behavior technique, using unsupervised and deep learning techniques, applied to historical EV data to make the day-ahead park.	K-means for clustering, multilayer perceptron for classification	More than 4 years data of the UCLA SMERC smart charging network infrastructure	Identified 4 clusters, with 3 relatively predictive behaviors, but one cluster represented random traveling schedule and energy consumption.
Kriekinge et al. [7] Brussels, Belgium 2023	Proposes a methodology to simulate charging demand for different EV driver types. The identification of similar profiles is performed using clustering.	K-means for clustering and KDE to better capture details for the simulation stage	8 755 private EV charging sessions (Jul 2018 - Jan 2022)	Identified 5 clusters, with distinct and different characteristics, showing good clustering results.
Gerossier et al. [8] Texas, USA 2019	Models the consumption profile of EVs from raw power measurements. The charging habits model is then used for forecasting short-term (1 day ahead) and long-term (2030).	Hierarchical clustering with Ward's method	46 private EV charging data recorded every minute of the year 2015 in Texas	Identified 4 clusters. Simulating the projected demand in 2030, it appears that the growth in EVs will have little effect on the load curve's shape.

3 Clustering algorithms

This section briefly describes the clustering algorithms that were used in the analysis of the EV charging profiles. Based on similarity measure, their goal is to partition the given dataset into intrinsic subgroups such that records within the same cluster are as similar as possible, while records belonging to different clusters are as dissimilar as possible. Cluster analysis operates with minimal prior information, utilizing an unsupervised learning approach that requires no training dataset for determining model parameters. This methodology serves as a cornerstone in exploratory data analysis, constituting a widely employed statistical technique applied across diverse domains.

Various types of clustering algorithms have been proposed in the literature: representative-based, hierarchical, density-based and graph clustering. In our analysis, we consider established algorithms from each type.

More specifically, **representative-based clustering** aims to divide a dataset into a predetermined number of clusters *k*. Each cluster is characterized by a representative record (called *centroid*), commonly chosen as the mean of within-cluster records, assuming that each record is modelled as a vector of numeric dimensions. The following algorithms are the main instantiations of this approach:

- **K-means** [9] is a greedy technique that conducts *hard clustering*, meaning that each record is assigned to just one cluster, i.e., the resulting clusters are disjoint. In essence, it partitions the input data to a predetermined number of clusters by assigning each record to its nearest centroid. It is described in more detail in Section 3.1.

- **Expectation-Maximization (EM)** [10] generalizes K-means by modelling the input data as a mixture of normal distributions. Its objective is to iteratively find the maximum likelihood of the cluster parameters, i.e,. the mean and covariance matrix. It is a *soft clustering* algorithm that returns the probability of a point belonging to each cluster. EM lies at the core of the GMM approach, which is described in Section 3.2.

Hierarchical clustering creates a sequence of nested partitions, which can be visualized as a tree, also called *dendrogram*, indicating the merging process and the intermediate clusters. The highest level (root) of the tree places all records in the same cluster, whereas the lowest level (leaves) consists of singleton clusters, yielding a separate cluster per input record. If the desired number of clusters is known, one can graphically see the level at which *k* clusters exist. This approach is implemented in two fundamentally different ways [11]:

- **Agglomerative** clustering operates in a bottom-up manner: it starts with singleton clusters and, at each step, it merges (i.e., agglomerates) the most similar (i.e., closest) pair of clusters until the desired number of clusters has been found. This requires a definition of cluster similarity.
- **Divisive** clustering operates in top-down manner: it starts with the root of the dendogram, i.e., a single cluster containing all input records, and at each step, it recursively splits one of the clusters until reaching the leaves of the dendogram, i.e., the singleton clusters. In this case, it is required to decide, at each stage, which cluster to split and how to perform this operation.

Given that the application of divisive clustering is quite challenging in terms of time complexity, our analysis exclusively considers agglomerative hierarchical clustering, which is analytically described in Section 3.3.

Density-based clustering leverages the connectedness of records in the multi-dimensional space defined by their numeric features to find nonconvex clusters. In other words, it defines clusters based on the local density of records, rather than relying exclusively on their similarities, as in K-means or EM. The most popular algorithms of this type are Density-Based Spatial Clustering of

Applications with Noise (DBSCAN) [12] and Ordering Points To Identify Cluster Structure (OPTICS) [13]. This type of algorithms are more suitable for geospatial data, which are excluded from our analysis. For this reason, we do not consider any of these algorithms in this deliverable.

Graph clustering transforms the input data into a *similarity graph*, where the nodes correspond to records and the edges connect pairs of records with non-zero similarity. The weight of each edge indicates the similarity of its adjacent records. Graph clustering can be viewed as an optimization problem over a k-way cut in the similarity graph, with different objectives represented as spectral decompositions of various graph matrices, e.g., the adjacency or the Laplacian matrix [14]. The similarity graph can then be split into connected components after applying the optimized cut, with the resulting components forming the final clusters. In this analysis, we are interested in algorithms satisfying three requirements:

1. They are *partitioning*, i.e., they generate disjoint clusters.
2. They are *unconstrained*, i.e., they require neither the number of final clusters nor their diameter or any domain-specific parameter to be pre-determined.
3. Their sole configuration parameter is a *similarity threshold **t***, which defines the minimum edge weight (i.e., all edges with a weight lower than t are discarded, before applying the clustering approach).

These two requirements are satisfied by seven main graph clustering algorithms: Connected Components, Center, Merge Center, Ricochet SR, Correlation, Markov and Cut Clustering. We briefly describe their functionality in Sections 3.4-3.10.

3.1 K-means Clustering

The goal of algorithm K-means [9] is to find a clustering that minimizes the Sum of Squared Errors (SSE) score, which measures the accuracy or goodness of the clustering, defined as:

$$SSE(C)= \sum_{i=1}^{k} \sum_{x_j \in C_i} ||x_j - \mu_i||^2 \qquad (3.1)$$

where $x_j \in R^d$ is a record from a given dataset $D^{n \times d}$ and $\mu_i \in R^d$ is the centroid of the clusters C_i. The records are then iteratively assigned to new centroids based on how close they are. In each iteration, the centroids are updated based on the mean of the assigned records. The process repeats until the centroids stop changing, as this is determined by a threshold. K-means is typically run multiple times, with the run with the lowest SSE value being selected to report the final clustering. This happens because the method begins with a random selection of the initial centroids. The elbow method is typically used to determine the optimal number of clusters.

3.2 GMM Clustering

Given n recors x_j in a d −dimensional space, let **X** = $(X_1, X_2, \dots, X_d)$ be the vector random variable across the d −attributes, with x_j being a data sample from **X**. The EM algorithm [10] assumes that each cluster C_i is characterized by a multivariate normal distribution:

$$f(\boldsymbol{x} \mid \boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}) = \frac{1}{(2\pi)^{(d/2)} |\boldsymbol{\Sigma_i}|^{1/2}} \exp\left\{ -\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu_i})^T \boldsymbol{\Sigma_i}^{-1} (\boldsymbol{x} - \boldsymbol{\mu_i}) \right\} \qquad (3.2)$$

where the centroid of the cluster C_i *is* $\mu_i \in R^d$ and the covariance $\sum_i \in \mathbb{R}^{d \times d}$ are both unknown parameters and f(x|μ_i,$\sum_i$) is the probability density at x attributable to cluster C_i.

A Gaussian Mixture Model over all k clusters defines the probability density function of **X**, given as:

$$f(\boldsymbol{x}) = \sum_{i=1}^{k} f(\boldsymbol{x} \mid \boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}) P(C_i), \tag{3.3}$$

where the prior probabilities P(C_i) satisfy $\sum_{i=1}^{k} P(C_i) \ = 1$. Thus, the Gaussian Mixture Model is characterized by the mean μ_i, the covariance $\sum_i$, and the mixture parameters for each of the k clusters, written compactly as:

$$\boldsymbol{\theta} = \{\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, P(C_i), \ldots, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, P(C_k)\}. \tag{3.4}$$

In this context, the goal of EM is to find the maximum likelihood estimates for the parameters θ.

During the Expectation Step, EM computes the cluster posterior probabilities through the Bayes theorem based on the current estimates for θ:

$$w_{ij} = P(C_i|\boldsymbol{x}_j) = \frac{P(\boldsymbol{x}_j|C_i)P(C_i)}{\sum_{a=1}^{k} P(\boldsymbol{x}_j|C_a)P(C_a)} = \frac{f_i(\boldsymbol{x}_j)P(C_i)}{\sum_{a=1}^{k} f_a(\boldsymbol{x}_j)P(C_a)}, \tag{3.5}$$

since each cluster is modeled as a multivariate normal distribution [15]. Therefore, P($C_i|x_j$) can be considered the weight contribution of x_j to cluster C_i .
Next, in the Maximization Step, EM recalculates θ using the weights w_{ij} .

3.3 Agglomerative Hierarchical Clustering

This algorithm starts with each of the *n* given records in a separate cluster. Then, the two closest clusters are repeatedly merged until all points are members of the same cluster. Given a set of clusters $C = (C_1, C_2, \ldots, C_m)$ first, the closest pair of clusters C_i and C_j are found and merged into a new cluster, C_{ij} . Next, the set of clusters is updated, removing C_i and C_j and adding C_{ij}. This process is repeated until Ccontains exactly k clusters.
Finding the closest pair of clusters is a key step in this procedure, which can leverage a variety of distance measures [16]. The main approaches in the literature are the following:

- **Single link**, where the distance between two clusters is defined as the minimum distance between a record in C_i and a record in C_j. First developed by Florek et al. [17] and then independently by McQuitty (1957) and Sneath (1957) [18];
- **Complete link**, where the distance between two clusters is defined as the maximum distance between a record in C_i and a record in C_j. Developed by Sørenson in 1948 [19];
- **Average link**, where the distance between two clusters is defined as the average pairwise distance between record in $C_i$and C_j. Developed by Sokal and Michener (1958) [20] to avoid the extremes introduced by either single or complete link;
- **Mean distance**, where the distance between two clusters is defined as the distance between the centroids of the two clusters. The earliest known use of this strategy is that of Sokal and Michener (1958) [20].
- **Ward's Method,** introduced by Joe H. Ward, Jr. in 1963 [21], where the distance between two clusters is defined as the increase in the sum of squared errors when the two clusters are merged.

The differences between these approaches are illustrates in *Figure 1*.

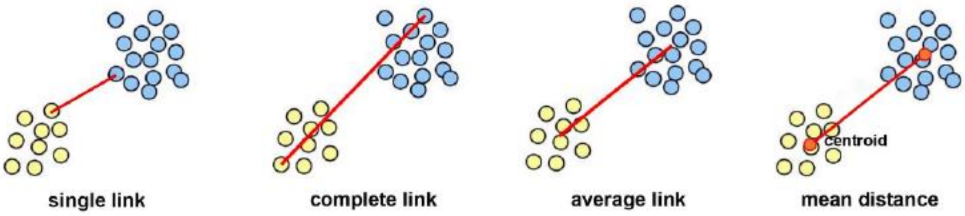


Figure 1. Different distance measures for Agglomerative Hierarchical clustering (Adapted from [22]).

Typically, the last option is applied in practice, which constitutes a weighted version of the mean distance as it weights the distance between centroids by half of the harmonic mean of the cluster sizes. Through Ward's method, Agglomerative Hierarchical Clustering minimizes the overall within-cluster variance.

3.4 Connected Components Clustering

This is the simplest graph clustering algorithm, as it simply computes the transitive closure of the pruned similarity graph, after discarding the edges with weight lower than the given threshold. An example of how this algorithm works is shown in *Figure 2:*

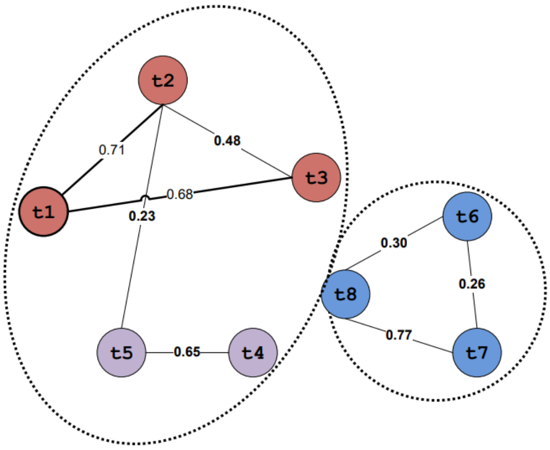


Figure 2. Applying connected component clustering.

3.5 Center Clustering

Center Clustering algorithm [23] partitions the similarity graph into clusters that have a centre, such that all records in each cluster are similar to the centre of the cluster. It operates as follows:

All edges (i.e., record pairs) are sorted in decreasing edge weight (similarity).
It iterates once over the sorted edges, starting from the top of sorted list.
For each edge (e_i, e_j), e_i is set as the center of the cluster if it is encountered for the first time.
Every record that appears in the subsequent pairs (e_i, e_n) or (e_n, e_i) is placed in the cluster of e_i.

Typically, this approach yields more clusters than Connected Components Clustering, because it assigns to a cluster only those records that are similar to the centre of the cluster. This is illustrated in *Figure 3*, which applies this algorithm to the same similarity graph as in *Figure 2.*

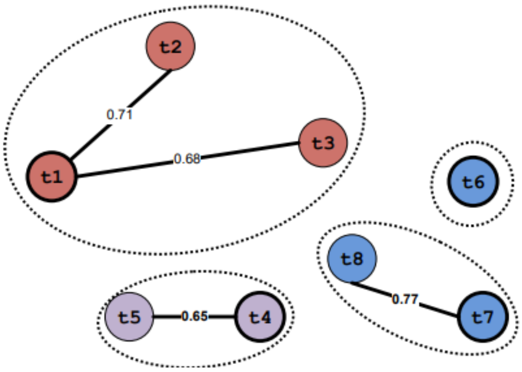


Figure 3. Applying center clustering.

3.6 Merge Center Clustering

Merge Centre Clustering algorithm [24] extends Center Clustering by merging two clusters c_i and c_j whenever a record similar to the center node of c_j is also similar to the center of c_j. This is accomplished efficiently, through a single iteration over the similarity graph edges, in the following way:

Similar to Center clustering, all edges are initially sorted in decreasing weight.
For every edge (e_i, e_j), e_i is set as the cluster center if it is encountered for the first time.
All subsequent edges (e_i, en) are placed in the cluster of e_i, if e_n is not associated with another cluster. Otherwise, the clusters of e_i and e_n are merged.

Inevitably, this approach yields less clusters than Center Clustering, but more clusters than Connected Components Clustering. This is illustrated in *Figure 4*, which applies to the same similarity graph as in *Figure 2* and *Figure 3*.

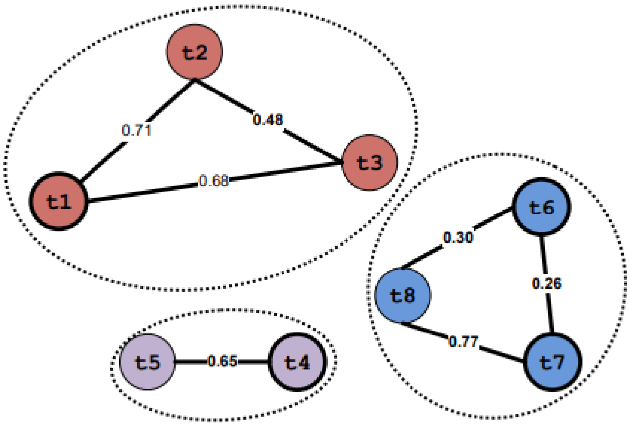


Figure 4. Applying Merge-Center Clustering.

3.7 Ricochet SR Clustering

Wijaya and Bressan [25] recently proposed a family of unconstrained algorithms called ‘Ricochet’. **Ricochet Clustering** is a family of algorithms whose strategy resembles the rippling of stones thrown in a pond. Their functionality combines ideas from classic K-Means and Star Clustering by alternating between two phases: first, they select seeds (star centers) for the clusters and then they refine the clusters iteratively. In this analysis, we consider **Sequential Rippling (SR)**, the most efficient algorithm of this family that generates disjoint clusters. It operates as follows:

For every node, the average edge weight in its neighborhood is computed.

```
All nodes are sorted in decreasing average edge score.
Starting from the top of the list, the next node $n_i$ is used as a seed for creating a new cluster, adding all neighboring nodes in it, as long as $n_i$ is not placed already in another cluster.
    All neighboring nodes are re-assigned to the new cluster, if they are closer to $n_i$ than to the seed of their current cluster.
    No cluster is created for $n_i$ if there are no node re-assignments.
    Seeds left with an empty cluster after re-assignment are moved to the cluster of their most similar neighboring node.
Terminate when all nodes have been assigned to a cluster.
```

3.8 Correlation Clustering

Correlation Clustering is a NP-hard problem [26]. This algorithm is crafted for signed undirected graphs, where every edge is labelled with + or −, depending on whether the adjacent nodes are similar or dissimilar. Its goal is to partition every signed graph in clusters that agree as much as possible with the edge labels. To this end, it solves a optimization problem where the goal is to find a partition that maximizes the number of + edges within clusters and the number of − edges between clusters (this is equivalent to minimizing the number of − edges inside clusters and the number of + edges between clusters). Given that this optimization task is an NP-hard problem, we consider an approximation solution with polynomial time complexity.

3.9 Markov Clustering

Markov Clustering, proposed by Stijn van Dongen [27], is an algorithm based on the simulation of stochastic flow in graphs. Its core assumption is that many edges within a region indicate a strong flow that gives rise to a separate cluster. In contrast, few edges and thus weak flow exists between such regions/clusters. To detect such regions, random walks over the entire graph are used to strengthen the flow where it is already strong, while weakening it where it is already weak. The random walks are terminated as soon as this structure emerges, forming regions with strong internal flow that are separated by boundaries with hardly any flow.

The time complexity of this approach is high. To enhance its scalability, we consider an optimized implementation, which maps the graph to a Markov matrix and recomputes the transition probabilities between nodes through the alternate application of two algebraic operations on matrices: expansion, which models the spreading of the flow through the normal matrix multiplication of a random walk matrix, and inflation, which models the contraction of the flow a Hadamard power followed by a diagonal scaling of another random walk matrix.

3.10 Cut Clustering

Cut clustering, proposed by Flake, Tarjan, and Tsioutsiouliklis [28], is an algorithm which partitions the similarity graph by discarding edges whose sum of weights is minimized, thus corresponding to edges between clusters. In the resulting pruned graph, the intra-cluster weights are maximized, indicating strong connections between the nodes of each cluster, while the inter-cluster weights are minimized. This approach is implemented by leveraging algorithms solving the maximum flow problem. To this end, an artificial sink node *s* is added to the similarity graph, as shown in *Figure 5*, which uses the same similarity graph as the ones presented in *Figure 2*, *Figure 3* and *Figure 4*.

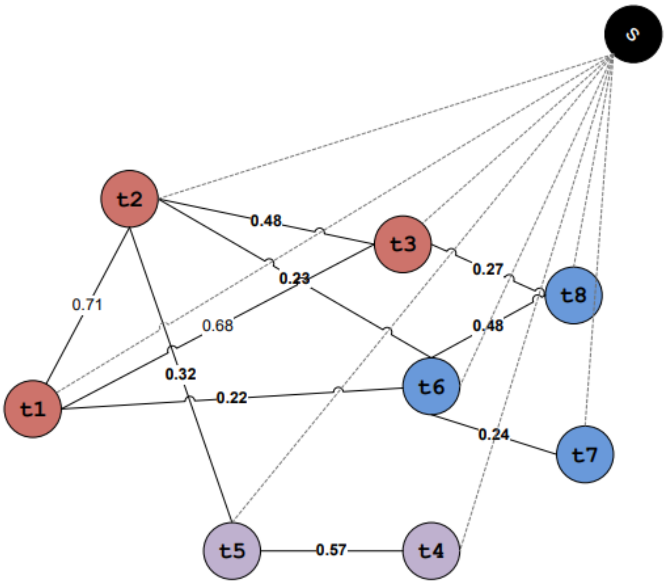


Figure 5. The artificial sink node added to the similarity graph by Cut Clustering.

In this extended similarity graph, the goal is to find the minimum cut between each node and *s*. The minimum cuts are iteratively detected, yielding a minimum cut tree. After removing the edges of the tree along with the sink *s*, the resulting connected components are the clusters of the original similarity graph.

4 Preliminary data analysis

The data used in this analysis was provided by PPC. With a total of 22,412 charging sessions, this dataset is one of the most complete ones available in the charging event format. The data was collected between July 2021 and May 2022 from public EVSEs in Greece, mainly located in high-traffic and quick-stay areas such as highways, gas stations, supermarkets, and stores. There are a total of 312 EVSEs with registered sessions in the dataset, of which only eight are of Level 3: six EVSEs with 50 kW, one with 60 kW, and one with 120 kW. All other chargers have 22 kW maximum power.

Figure 6 presents the monthly and weekly progression of session numbers, from which one sees an increase until the end of November 2021, reaching a maximum around this time. However, there was a sharp decline in December 2021 and January 2022, coinciding with Greece's highest peak of COVID 19 cases due to the Omicron variant. The session numbers remain consistent throughout the rest of 2022, exhibiting an increasing trend.

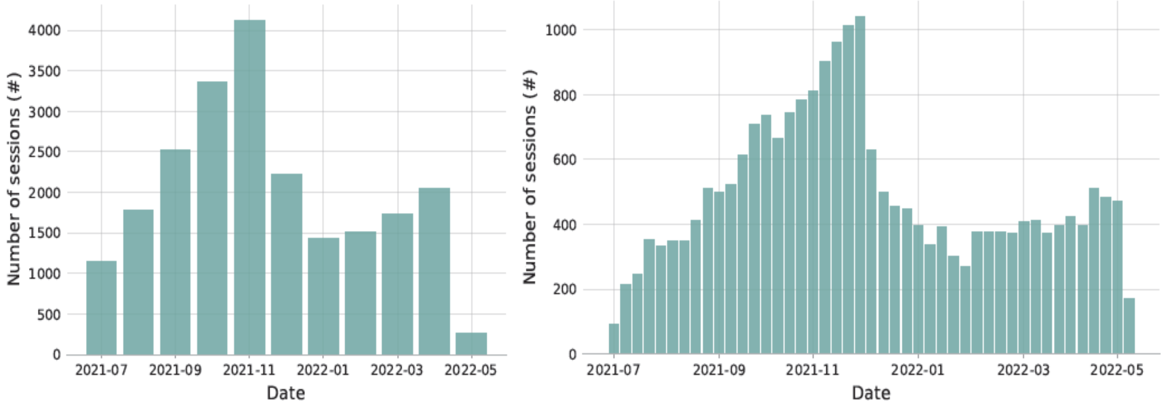


Figure 6. Monthy (left) and weekly (right) charging activity in the dataset.

The original data involves the following fields:

1. Start datetime
2. End datetime
3. Volume (kWh delivered)
4. Charge_Point_Address
5. Charge_Point_ZIP
6. Charge_Point_City
7. Charger ID
8. Max power EVSE
9. Authentication ID
10. Sojourn time

Below, we describe the process that we applied in order to improve the quality and content of the raw data and to bring it into a format suitable for clustering. From our analysis, we exclude the site specific fields, namely Charge_Point_Address, Charge_Point_ZIP and Charge_Point_City.

4.1 Dealing with noisy and missing data

Some records have missing information, such as the plug-in/plug-out times or the energy consumed. These empty values prevent the correct implementation of clustering methods. There are two main ways of addressing this kind of noise: *(i)* Interpolation, where nearby entries are used to replace these absent values; *(ii)* Removal, where records with missing entries are excluded from the analysis, resulting in a dataset with real, accurate and unaltered data. We have opted for the latter approach, to avoid records with artificial values in some of their fields.

When removing the records with missing values, we noticed that this primarily applies to the field Max Power EVSE, whose omission makes it impossible to determine the average charging and idle time (see below for more details).

Moreover, we removed noise from records with an average charging time higher than the sojourn time, which indicates that the EV was effectively charging during the entire parking period and that the adjustment factor was too harsh for these particular sessions. In these records, Average Charging Time (see Section 4.3) was set equal to Sojourn Time, leading to a zero idle time.

4.2 Outlier detection and removal

Another form of noise comes from outliers, i.e., records with inaccurate information, such as an abnormal energy supply, or EV drivers with an excessive number of sessions. Outliers should be detected and removed using, for instance, techniques like Interquartile Range, Elliptic Envelope, Isolation Forest, or by defining thresholds for data removal. Due to the low number of fields per record, we opted for the last approach, which allows for leveraging our domain knowledge to achieve higher accuracy.

More specifically, to remove outliers, we defined thresholds for specific fields:

- The charging and sojourn time cannot exceed 24 hours (there are just 31 records with a parking stay longer than this limit).
- Only sessions with more than 1 minute of sojourn time are considered.
- The maximum energy delivered cannot exceed 100 kWh considering the 2022 EV sales in Europe.

Finally, all records with negative values for at least one field were also removed.

Together with the previous step, noise removal, this step yields the clean dataset which comprises:

- 21,801 records/charging sessions
- 3,184 different users
- 313 different charging points.

4.3 Feature engineering

The goal of this step is to define features not previously included in the dataset that help to analyze and obtain more meaningful clusters. More specifically, two time periods in EV charging are important:

- the time (t) the EV was parked and plugged into the EVSE (*Sojourn Time*), and
- the fraction thereof that is effectively spent on charging (*Charging Time*).

With these two indicators, the Idle Time can be determined, as a measure of flexibility of the charging process. More formally, these new features can be defined as:

$$\begin{gathered}\text{Sojourn Time} = \mathrm{t}^{plug-out} - \mathrm{t}^{plug-in}\\ \text{Charging Time} = \mathrm{t}^{endcharging} - \mathrm{t}^{startcharging}\\ \text{Idle Time} = \text{Sojourn Time} - \text{Charging Time}\end{gathered} \tag{4.1}$$

The dataset already provides the sojourn time, but does not provide the session's end of charging, and consequently the above definition of Charging Time cannot be employed. Instead, it offers information on the maximum power capacity of the EVSEs. As a result, it is possible to obtain an estimated value of the charging time for each session through the following formula:

$$\text{Average Charging Time}_{\text{sessioni}} = \frac{EnergyDelivered_i}{(maxPowerEVSE)_i \times AdjustmentFactor} \tag{4.2}$$

An adjustment factor (set equal to 0.8 in our analysis) guarantees a realistic charging time, since this process is not carried out at a constant power rate; it depends on external factors such as temperature, high loads on the grid, and the state-of-charge (as the battery becomes fully charged, the charging rate decreases). Thus, this factor ensures a 20% safety margin for the maximum power value.

Note that to infer the idle time from the sojourn and charging time, their DateTime values were transformed into float format as follows (using the DateTime method of Pandas library[2]): the value was converted to seconds and then divided by 3,600 to get the hour of the day in decimal form. For example, 10h17 (10 hours and 17 minutes) becomes 10.28h (10.28 hours). This numeric form allows for applying outlier removal approaches, clustering methods, and graphical representations.

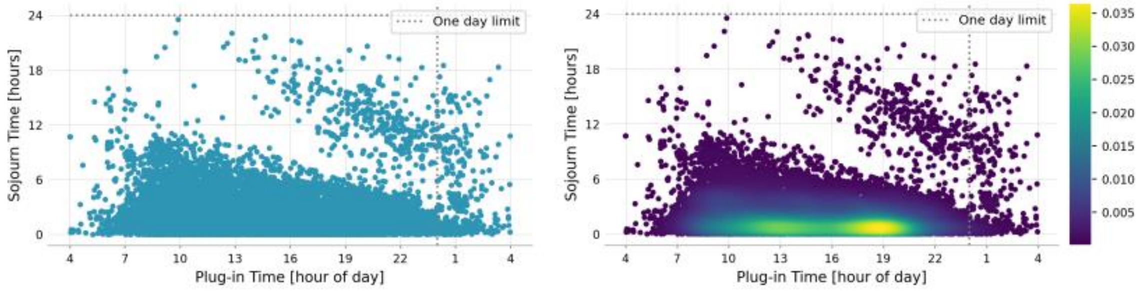
4.3.1 Data Adjustment

The above changes reduced the originally 22,412 records to 21,801. The final fields per record, all in float format and free of missing values, are the following:

1. Start datetime
2. End datetime
3. Volume (kWh delivered)
4. Charger ID
5. Max power EVSE
6. Authentication ID
7. Sojourn time
8. Average charging time
9. Idle time

Note that fields 1 and 2 originally contained both the day and the corresponding hour, but were converted to just hours of the day when the DateTime variables were converted into float values. As a result, the time frame under consideration became 00h00 to 23h59, losing the spatial proximity of early and late plug-in times. Given that there is an instant around 04h00, when charging activity is at its lowest, we restored the spatial proximity in the dataset records by relocating all charging sessions with plug-in times less than this minimum to the right side to continue the timeframe after 23h59. The final clean and pre-processed dataset is illustrated in *Figure 7*.

[2] https://pandas.pydata.org



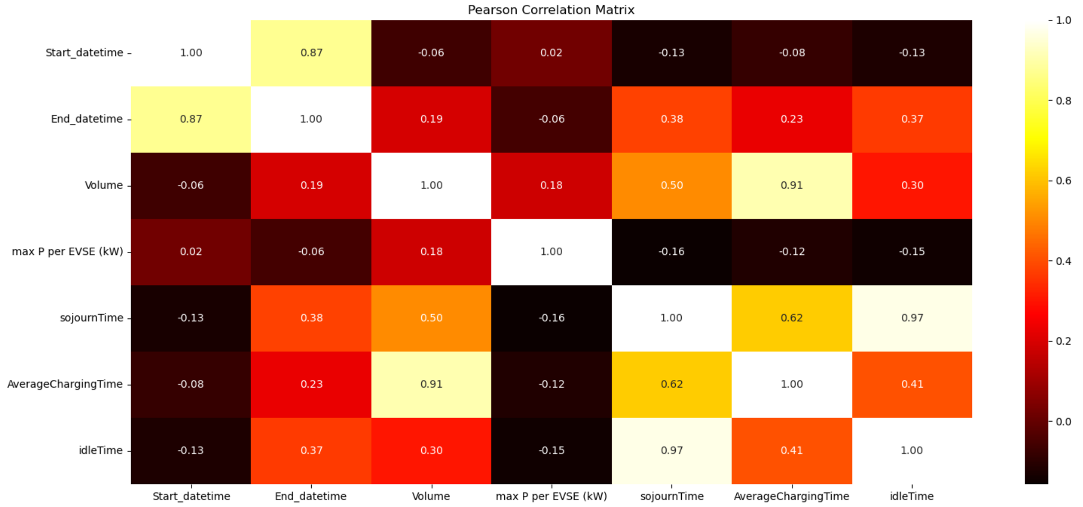
(a) Scatter plot. **(b)** Density Scatter plot.

Figure 7. Final adjusted distribution regarding Sojourn Time and Plug-in Time.

Finally, another indispensable step is data **normalization**. Clustering algorithms are sensitive to the scale of the data, because they involve distances, densities, or both; if the considered features have different scales, some fields inevitably dominate others. Normalizing the data ensures that each entry contributes equally to the distance calculation between data points, helping to improve the accuracy of the clustering algorithms and generate good-quality clusters. Consequently, each dataset field (column) should range from 0 to 1, allowing an overall normalization of the data. To achieve this, we applied the **MinMaxScaler** method from the scikit-learn[3] Python library.

4.4 Feature selection

To improve the quality of clustering, we need to retain the absolutely necessary features, ensuring that there are no correlations between them. To this end, we computed the correlation matrix, which is shown in *Figure 8*. Note that every cell indicates the Pearson correlation between the values of the corresponding fields; high positive or high negative values indicate that they two fields follow the same patterns, i.e., when the value in one field increase, the value of the other increases or decreases, respectively, to a similar extent. Note also that we have excluded the Authentication and Charger Id, as they both take random numbers or even non-numeric signatures as values.


Pearson Correlation Matrix

	Start_datetime	End_datetime	Volume	max P per EVSE (kW)	sojournTime	AverageChargingTime	idleTime
Start_datetime	1.00	0.87	-0.06	0.02	-0.13	-0.08	-0.13
End_datetime	0.87	1.00	0.19	-0.06	0.38	0.23	0.37
Volume	-0.06	0.19	1.00	0.18	0.50	0.91	0.30
max P per EVSE (kW)	0.02	-0.06	0.18	1.00	-0.16	-0.12	-0.15
sojournTime	-0.13	0.38	0.50	-0.16	1.00	0.62	0.97
AverageChargingTime	-0.08	0.23	0.91	-0.12	0.62	1.00	0.41
idleTime	-0.13	0.37	0.30	-0.15	0.97	0.41	1.00

Figure 8. Correlation matrix of the fields in the cleaned dataset.

[3] https://scikit-learn.org

By analyzing the covariance matrix between the remaining features, interesting patterns arise. The highly correlated "Start datetime" and "End datetime" are redundant, so only one is required. The former provides more intelligible information, and thus it was eventually chosen. The same reasoning applies to "Volume" (i.e., kWh delivered) and "Average Charging Time". Among the two, we chose Volume. The Sojourn Time strongly correlates with "Idle Time", which is thus redundant, as expected. Therefore, we retain the following three complementary, non-redundant features that align with the current objective:

1. Start datetime
2. Volume (kWh delivered)
3. Sojourn time

5 Methodology

The overview of the methodological approach we applied in this analysis is illustrated in Figure 9.

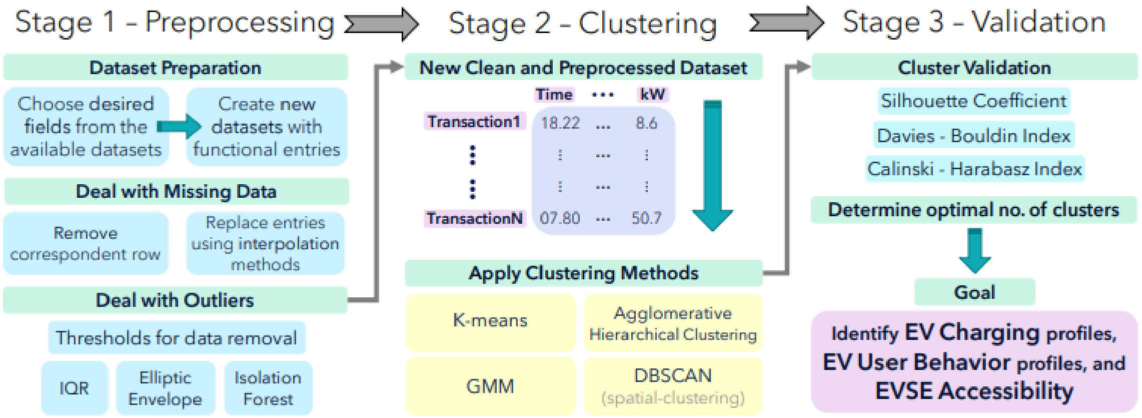

Stage 1 - Preprocessing
Stage 2 - Clustering
Stage 3 - Validation
Dataset Preparation
Choose desired fields from the available datasets
Create new datasets with functional entries
Deal with Missing Data
Remove correspondent row
Replace entries using interpolation methods
Deal with Outliers
Thresholds for data removal
IQR
Elliptic Envelope
Isolation Forest
New Clean and Preprocessed Dataset
Time
kW
Transaction1
18.22
8.6
TransactionN
07.80
50.7
Apply Clustering Methods
K-means
Agglomerative Hierarchical Clustering
GMM
DBSCAN (spatial-clustering)
Cluster Validation
Silhouette Coefficient
Davies - Bouldin Index
Calinski - Harabasz Index
Determine optimal no. of clusters
Goal
Identify EV Charging profiles, EV User Behavior profiles, and EVSE Accessibility

Figure 9. Overview of our methodological approach.

Stage 1 focuses on Preprocessing, which is described in Section 4. Next, Stage 2 applies the clustering algorithms described Sections 3.1-3.10 to the resulting clean data to identify groups of similar charging patterns and EV user behavior. For each algorithm, the most crucial step is to fine-tune its configuration parameters. For all algorithms, this is carried out with grid search over a reasonable set of configuration values. This process is analytically described in Section 5.1. Due to the lack of ground-truth, i.e., the cluster that is ideally associated with each record, the evaluation of the resulting clusters is based on three measures that leverage the intra- and inter-cluster characteristics, i.e., on commonalities between records of the same cluster and between records across different clusters. These evaluation measures are presented in Section 5.2.

5.1 Configuration Parameter Fine-tuning

The clustering algorithms we are considering in this analysis are distinguished into two main categories, according to the configuration parameter that needs to be fine-tuned:

- Those depending on the number of clusters ***K***.
- Those depending on the threshold ***t*** pruning the low-weighted edges of the similarity graph.

The former category includes K-Means Clustering, GMM and Agglomerative Hierarchical Clustering. To configure the required number of clusters, we apply the **elbow method**: we try all values from 2 to 18 with a step of 1. For each K, we compute **inertia**, also known as *within-cluster sum-of-squares*, which amounts to the sum of the squared distances of all records from their closest centroid. A two-dimensional plot is formed with K on the horizontal axis and inertia on the vertical one. Typically, the resulting curve starts from the upper left corner and ends at the lower right one, as in *Figure 10(a)*. The best value for K is determined as the one corresponding the “elbow” or “knee” of this curve, where the rate of reduction is significantly reduced for the first time.

The latter category of methods includes all graph clustering techniques which are presented in Sections 3.4-3.10. Their similarity threshold takes all values from 0 to 1 with a step of 0.1. For each of these values, we compute the number of resulting clusters. Apparently, thresholds yielding a single cluster are disregarded from further analysis. For the remaining threshold values, we compute the **entropy** of the distribution of cluster sizes. This approach favours thresholds generating few clusters of balanced size over thresholds generating a large number of clusters with a high diversity in their sizes. Finally, we select as the optimal similarity threshold for each graph clustering algorithm the one maximizing the entropy of cluster sizes.

5.2 Clustering Validation Techniques

As in any clustering task, there is no ground truth available. As a result, the quantitative evaluation of the clustering algorithms relies on three internal validation metrics, which consider the intra- and inter-cluster characteristics: the Silhouette coefficient[29], the Davies-Bouldin index[30], and the Calinski-Harabasz index[31]. We formally define them below.

5.2.1 Silhouette Coefficient

For each record x_i , the silhouette coefficient is formally defined as:

$$s_i = \frac{\mu_{out}^{min}(\boldsymbol{x}_i) - \mu_{in}(\boldsymbol{x}_i)}{\max\{\mu_{out}^{min}(\boldsymbol{x}_i), \mu_{in}(\boldsymbol{x}_i)\}}, \tag{5.1}$$

where $\mu_{out}^{min}(x_i)$ is the mean of the distances from x_i to records in the closest cluster, and $\mu_{in}(x_i)$ is the mean distance from x_i to records in its own cluster.

The total Silhouette coefficient is defined as the mean s_i value across all records:

$$SC = \frac{1}{n} \sum_{i=1}^{n} s_i \tag{5.2}$$

SC takes values from -1 to +1, with higher values indicating more precise clustering, i.e., most records are well matched to their own clusters and poorly matched to neighboring ones.

5.2.2 Davies-Bouldin Index

The Davies-Bouldin measure for a pair of clusters C_i and C_j is formally defined as:

$$DB_{ij} = \frac{\sigma_{\mu_i} + \sigma_{\mu_j}}{\delta(\mu_i, \mu_j)} \tag{5.3}$$

where μ_i denotes the centroid of cluster C_i, $\sigma_{\mu i} = \sqrt{var(C_i)}$ represents the dispersion of the records around the respective centroid (i.e., the square root of the total variance) and $\delta(\mu_i, \mu_j)$ is the distance between the centroids.

The Davies-Bouldin index is thus defined as:

$$DB = \frac{1}{k} * \sum_{i=1}^{k} \max_{i \neq j}\{DB_{ij}\} \tag{5.4}$$

This means that for each cluster C_i, only the cluster C_j with the largest DB_{ij} measure is considered. Therefore, smaller DB values, closer to zero, mean better performance, as clusters are well separated and each one is well represented by its centroid.

5.2.3 Calinski-Harabasz Index

Given the dataset D=$\{x_i\}_{i=1}^{n}$, the Calinski-Harabasz index is formally defined as:

$$CH(k) = \frac{tr(S_B)}{tr(S_W)} * \frac{n-k}{k-1} \tag{5.5}$$

where $tr(S_B)$ denote the trace of the within-cluster scatter matrix, and $tr(S_w)$ stands for the trace of the between-cluster scatter matrix. Those matrices are defined by the equations below, respectively, where μ is the dataset's mean and μ_i is the mean for cluster C_i.

$$\begin{gathered} S_B = \sum_{i=1}^{k} n_i(\mu_i - \mu)(\mu_i - \mu)^T \\ S_W = \sum_{i}^{k} \sum_{x_j \in C_i} (x_j - \mu_i)(x_j - \mu_i)^T \end{gathered} \tag{5.6}$$

A good number of clusters k should result in a high $CH(k)$. The intuition is to determine the value of k for which $CH(k)$ is higher than $CH(k-1)$ and there is a slight improvement or a decrease in the $CH(k+1)$ value. This way, the Calinski-Harabasz index can be also used to choose the number of clusters that maximize $CH(k)$, an alternative to the elbow method we described in Section 5.1.

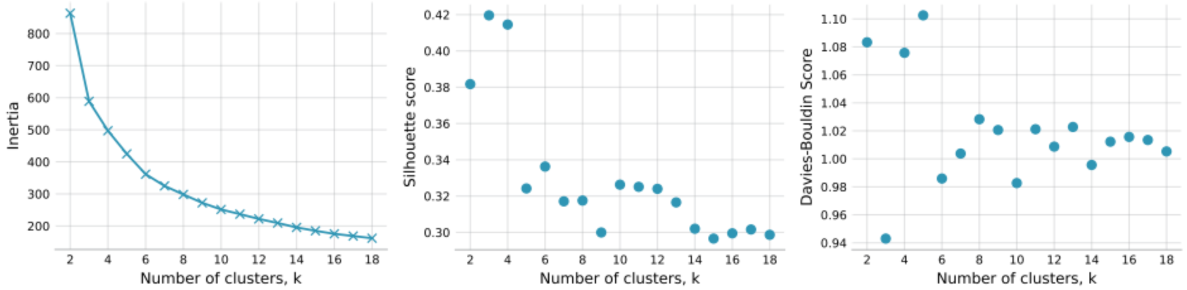
6 Results

We now describe the results produced by the clustering algorithms discussed in Sections 3.1-3.10. For K-Means, GMM and Agglomerative Hierarchical Clustering, we used Python 3.10 and the **scikit-learn**[4] library in particular. For the graph clustering algorithms, we used **FAISS**[5] in order to quickly compute all pairwise similarities and **pyJedAI**[6] for the implementation of all algorithms described in Sections 3.4-3.10. However, the similarity graph is a complete one with 21,801 × 21,800 / 2 = 237,6 million edges, with the ensuing space requirements exceeding the available memory (64 GB). To address this issue, we used the corresponding Java implementation, which is available through **JedAI**[7].

Below, we delve into the results of each clustering algorithm.

6.1 K-means Clustering

The number of clusters, *k*, was chosen based on the elbow method and the values obtained for the Silhouette, Davies-Bouldin, and Calinski-Harabasz scores, also considering the resulting profiles. *Figure 10* illustrates the plots of the different scores as a function of k, which increases from 2 to 18 with a step of 1. The elbow method does not effectively display an elbow, making it insufficient for determining the ideal k. Nevertheless, the knee of the curve suggests that the ideal value for k fluctuates between 5 and 8. Within this range, by performing a more in-depth analysis, the best results are thus found for k=6, with better Silhouette and lower Davies-Bouldin scores, as shown in *Figure 10*(b) and *Figure 10*(c), respectively. Note that we have excluded the Calinski-Harabasz score, because it behaves similarly to inertia in *Figure 10*(a). Both plots, though, indicate a turning point at k=10, with interesting scores compared with the remaining k's.



(a) Elbow Method. **(b)** Silhouette Coefficient. **(c)** Davies-Bouldin Index.

Figure 10. Different scores as a function of k for the K-means clustering.

To select between k=6 and k=10, we performed a qualitative analysis that indicated more useful results in the latter case. More specifically, k=6 yields quite generic profiles that comprise relatively different behaviors within the same clusters, whereas k=10 generates clusters that are better defined and identifiable. This is illustrated in *Figure 11*, which presents the distribution of the adjusted EV charging profiles for k=10 based on the selected fields, i.e., "Start datetime", "Sojourn Time", and "Volume" (kWh delivered). There is, however, greater separation between the sessions as five

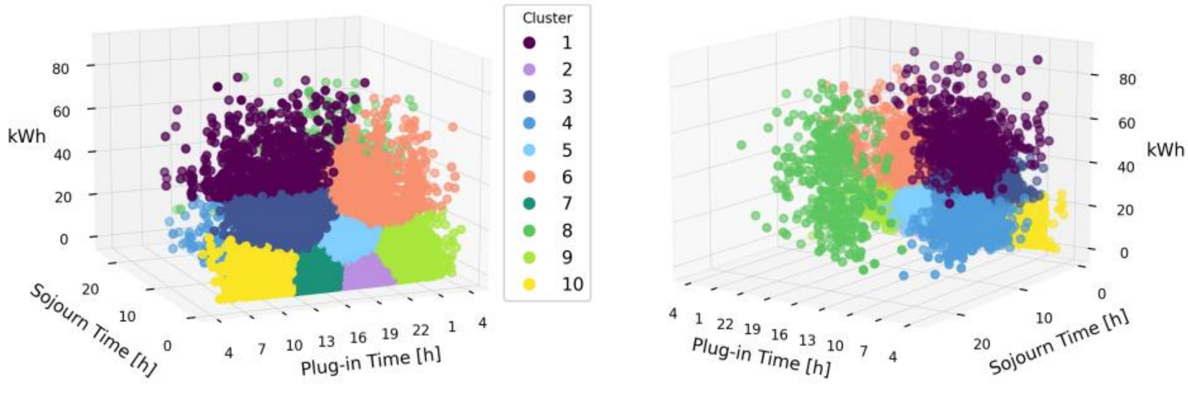
[4] https://scikit-learn.org

[5] https://faiss.ai/index.html

[6] https://pyjedai.readthedocs.io/en/latest/intro.html

[7] https://github.com/scify/JedAIToolkit

clusters were found with plug-in times in the morning (clusters 1, 3, 4, 7, and 10) and only three with plug-in times in the evening (clusters 6, 8 and 9). There are also two clusters during the middle/late afternoon (clusters 2 and 5). Therefore, one can conclude that the sessions during the day differ significantly from each other, translating into a high number of daily. Additionally, the reduced number of clusters in the evening suggests that the sessions during this period exhibit more similar behavior than those during the day.



(a) Azimuth = -115°.

(b) Azimuth = 125°.

Figure 11. 3D distribution of the K-means EV Charging profiles for k=10.

Another interesting point is that the most different sessions (higher sojourn times and, thus, higher flexibility potential) fall into distinct clusters: cluster 8 contains the sessions that only end the next day, regardless of the plug-in time, while cluster 4 comprises the sessions that start in the morning and only end in the afternoon of the same day. Table 2 lists the mean quantitative characteristics of the resulting ten profiles.

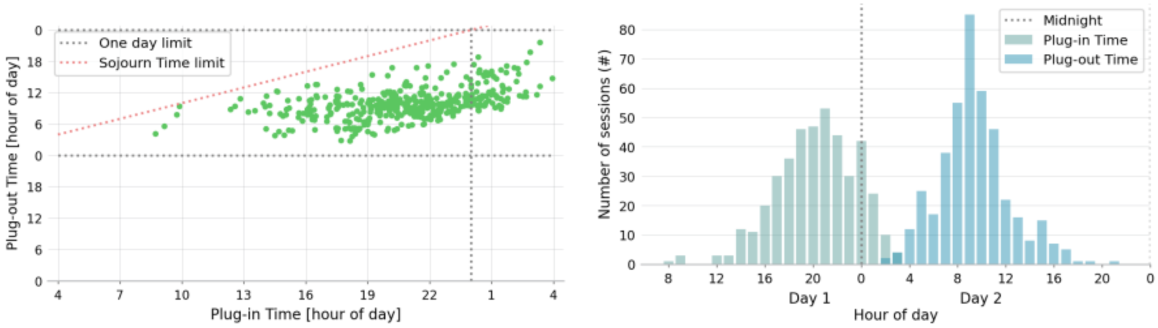
Table 2. Mean quantitative characteristics of the K-means EV Charging profiles for k=10.

Cluster ID	No. of Sessions	Plug-in Time	Plug-out Time	Energy [kWh]	Sojourn Time	Charging Time	Idle Time	Profile*
1	704	11h39	16h33	49,497	4h54	2h23	0h31	Morning to afternoon high energy, long-term stay
2	4,297	18h25	19h07	3,957	0h42	0h42	0h29	Early evening low energy, short-term stay
3	1,500	11h49	14h20	26,807	2h31	1h17	1h15	Early afternoon medium energy, medium-term stay
4	1,384	10h35	15h51	12,187	5h16	0h41	4h34	Morning to afternoon medium energy, long-term
5	2,520	17h05	19h18	14,759	2h13	0h45	1h28	Afternoon to evening medium energy, medium-term

6	1,154	19h58	22h43	35,992	2h45	1h35	1h10	Evening to night high energy, medium-term s
7	4,529	13h42	14h40	53,141	0h58	0h17	0h41	Early afternoon low energy, short-term s
8	419	20h43	09h51	33,445	13h08	1h54	11h14	Evening to next morning medium energy, long-term
9	1,888	21h45	23h06	9,788	1h21	0h29	0h52	Night low energy, shortterm stay
10	3,406	09h43	10h54	6,835	1h10	0h21	0h49	Morning low energy, shortterm stay

*Note: "Low energy": below 10 kWh; "Medium energy": between 10 kWh and 30 kWh; "High energy": over 30 kWh. "Short-term": sojourn time below 2h; "Medium-term": between 2h and 4h; "Long-term": over 4h.

According to Table 2, one verifies that clusters 2 and 7 are the most typical profiles, as they comprise the highest number of sessions, meaning that the short and low-energy sessions are the most frequent, and the later the drivers plug in, the more energy they consume. Morning and afternoon profiles generally involve lower energy delivery. Cluster 8 is better defined, since it exclusively contains sessions that end the next day. Consequently, the mean flexibility potential (idle time) of this profile is even greater, with more than eleven hours of parking stay without charging. To get a better perspective on this behavior, *Figure 12* illustrates the distribution of the corresponding sessions.



(a) Scatter plot of sessions.

(b) Distribution of sessions.

Figure 12. Deep examination of K-means GR-Data cluster 8, regarding the Plug-in and Plug-out time (i.e., Start and End datetime).

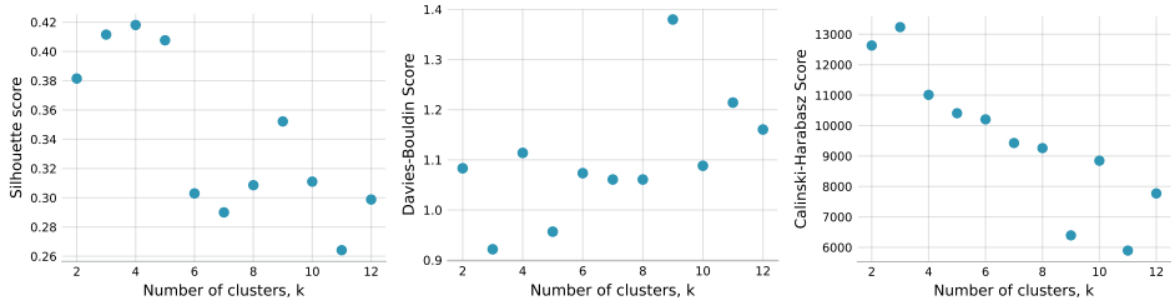
6.2 GMM Clustering

According to the scikit-learn, the GMM method includes four choices for the covariance type:

1. full covariance, where each component has its own overall covariance matrix,
2. tied covariance, where all components share the same overall covariance matrix,
3. diagonal covariance, where each component has its own diagonal covariance matrix, and
4. spherical covariance, where each component has its own unique variance.

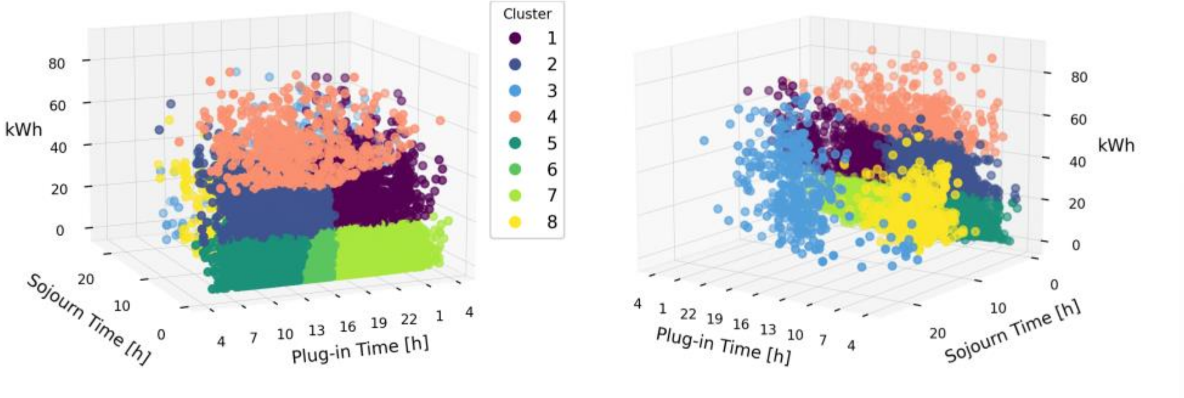
Consequently, in addition to the number of clusters, it was also necessary to understand which type of covariance provides the best profiles and scores. Thus, a preliminary cluster analysis proved that *tied covariance* originates meaningful profiles, achieving the best Silhouette, Davies-Bouldin, and Calinski-Harabasz scores, regardless of the number of clusters.

Figure 13 shows the performance of GMM with tied covariance with respect to the three evaluation measures. The obtained scores do not indicate a clear choice for the optimal k, with the best options being one of 6, 8 and 10. Further qualitative analyses revealed that k=8 is the best choice, yielding a good balance between scores and meaningfulness. *Figure 14* presents the distribution of the adjusted EV charging profiles regarding the Plug-in Time, Sojourn Time, and kWh.



(a) Silhouette Coefficient. **(b)** Davies-Bouldin Index. **(c)** Calinski-Harabasz Index.

Figure 13. Different scores of GMM Clustering with tied covariance as a function of k.



(a) Azimuth = -115°. **(b)** Azimuth = 125°.

Figure 14. 3D distribution of the GMM EV Charging profiles with k=8 and tied covariance.

Figure 14 reveals a clear division concerning the energy delivered: profiles up to 20 kWh (clusters 5, 6, and 7), up to 40 kWh (clusters 1 and 2), and finally above 40 kWh (cluster 4). In fact, contrary to K-means, GMM groups all the highest energy sessions in cluster 4, without differentiating the plug-in time. Although fewer in number, the clusters differentiate the sessions with higher sojourn time, namely clusters 3 and 8. Cluster 3 contains the sessions with the highest sojourn times, most of which do not finish until the following day. However, it also includes some sessions with a plug-in time of around 07h00 that end on the same day, unlike K-means cluster 8 (see *Figure 11*).

Table 3 lists the quantitative mean characteristics of the eight profiles, demonstrating that cluster 4, which has the highest energy delivered and relatively fast charging, contains few sessions and is the second least usual. Cluster 3 is the least common, corresponding to highly flexible night-time charging sessions.

Table 3. Mean quantitative characteristics of the GMM EV Charging profiles for k=8.

Cluster ID	No. of Sessions	Plug-in Time	Plug-out Time	Energy [kWh]	Sojourn Time	Charging Time	Idle Time	Profile*
1	1,241	20h01	23h01	33,414	2h59	1h35	1h24	Evening to midnight high energy, medium-term stay
2	1,521	11h57	14h58	31,430	3h01	1h32	1h29	Morning to afternoon high energy, medium-term stay
3	392	20h02	09h33	29,090	13h32	1h39	11h52	Evening to next morning medium energy, long-term
4	491	14h13	17h21	55,446	3h08	2h04	1h04	Afternoon high energy, medium-term stay
5	6,093	10h38	12h06	7,771	1h28	0h25	1h04	Morning low energy, shortterm stay
6	3,661	14h35	15h50	6,746	1h15	0h22	1h04	Afternoon low energy, short-term stay
7	7,724	19h04	20h16	7,397	1h12	0h23	0h48	Evening low energy, shortterm stay
8	678	10h56	18h01	15,367	7h05	0h52	6h13	Morning to evening medium energy, long-term

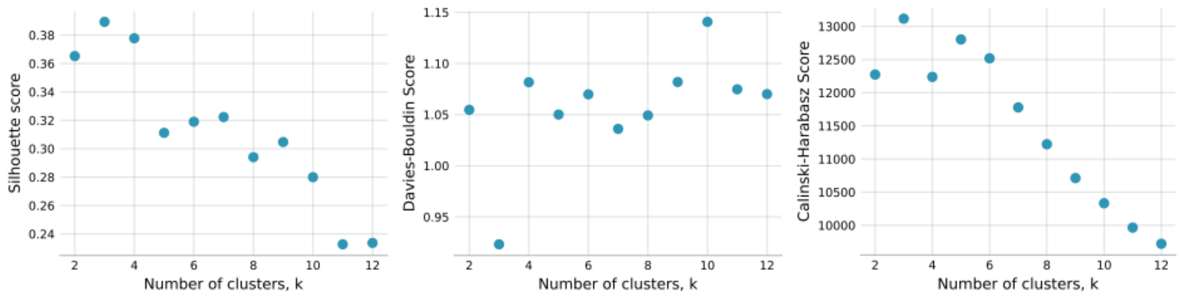
*Note: “Low energy”: below 10 kWh; “Medium energy”: between 10 kWh and 30 kWh; “High energy”: over 30 kWh. “Short-term”: sojourn time below 2h; “Medium-term”: between 2h and 4h; “Long-term”: over 4h.

As seen for K-means, the short duration and low energy profiles are the most frequent (clusters 5, 6, and 7). Clusters 3 and 8 offer significant flexibility potential due to their high idle times resulting from fast charging. This suggests that using such high charging power does not make sense since EV drivers tend to park longer than the car is effectively charging. Reducing the charging rate (maximum EVSE power) during those sessions would result in fewer power peaks on the grid.

6.3 Agglomerative Hierarchical Clustering

The configuration of this algorithm requires selecting one of the distance measures in Section 3.3. Preliminary experiments revealed that none of the four measures consistently outperforms all others in terms of the three evaluation measures for effectiveness. A further qualitative analysis, though, indicated that Ward’s method achieves the best balance between meaningful profiles and the evaluation measures.

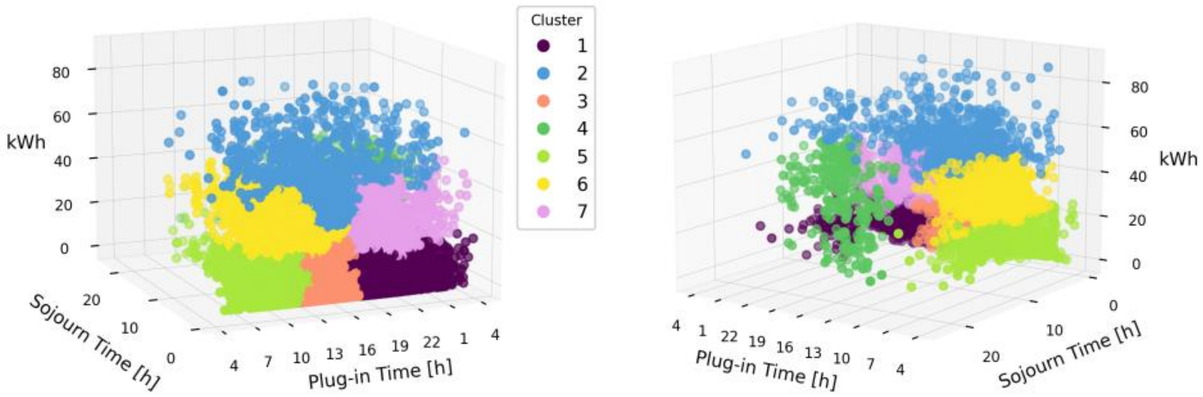
Using Ward’s distance, *Figure 15* illustrates the plots of the different scores as a function of k. According to the scores, it is clear that k=7 gives the highest Silhouette score and the lowest Davies-Bouldin score compared with the k’s immediately below or above. The Calinski-Harabasz score does not contribute to determining the number of clusters since it displays a hyperbolic behavior for k > 6.



(a) Silhouette Coefficient. **(b)** Davies-Bouldin Index. **(c)** Calinski-Harabasz Index.

Figure 15. Different scores as a function of k for the Agglomerative Hierarchical Clustering, with Ward's method as distance measure.

Figure 16 presents the distribution of the resulting EV charging profiles for k=7 regarding the Plug-in Time, Sojourn Time, and kWh fields, from which one sees that the profiles exhibit more overlap and less clear definition than those found with the K-means or GMM clustering. Despite several overlapped points with neighboring clusters and the absence of GMM cluster 8, the obtained profiles visually resemble the results of the GMM clustering. As a result, clusters 5 and 6 contain sessions with differing behaviors due to the grouping of short and long morning sessions. Furthermore, cluster 1 includes some sessions that end the following day, affecting the profile characterization. Overall, the clusters are poorly defined, resulting in less accurate and reliable identification of typical profiles.



(a) Azimuth = -115°. **(b)** Azimuth = 125°.

Figure 16. 3D distribution of the EV Charging profiles resulting from the Agglomerative Hierarchical Clustering with Ward's method as distance measure and k=7.

Table 4 lists the mean quantitative characteristics of the seven profiles. The results confirm that Hierarchical clustering produces significantly different results when compared with K-means and GMM. The clustering method did not differentiate short-term sessions, which were grouped with sessions of longer duration, as seen in clusters 5 and 6, for example. Cluster 4, typical of nighttime charging that only ends the next day, includes a reduced number of sessions, with a large part of these next morning sessions incorporated into clusters 1 and 2. Nevertheless, the average characteristic values of each profile are still relevant. We increased the number of clusters to solve these shortcomings, but higher k yielded new, meaningless clusters in terms of EV charging profiles. Thus, k=7 is effectively the best number of clusters for this method, but compared to the best configurations of K-means and GMM, it yields inferior results.

Table 4. Mean quantitative characteristics of the Agglomerative Hierarchical Clustering EV Charging profiles for k=7.

Cluster ID	No. of Sessions	Plug-in Time	Plug-out Time	Energy [kWh]	Sojourn Time	Charging Time	Idle Time	Profile*
1	6,515	19h25	20h38	6,750	1h13	0h21	0h51	Evening low energy, shortterm stay
2	913	15h09	19h15	50,102	4h06	2h11	1h55	Afternoon to evening high energy, long-term stay
3	6,172	14h20	15h39	6,711	1h19	0h21	0h58	Afternoon low energy, short-term stay
4	298	20h14	9h48	33,137	13h34	1h53	11h41	Evening to next-morning high energy, long-term stay
5	4,787	09h57	12h00	8,007	2h03	0h26	1h37	Morning low energy, medium-term stay
6	1,604	11h18	14h45	28,635	3h26	1h26	2h00	Morning to afternoon medium energy, medium-term
7	1,512	19h56	22h15	27,923	2h20	1h18	1h02	Evening to midnight medium energy, medium-term

*Note: "Low energy": below 10 kWh; "Medium energy": between 10 kWh and 30 kWh; "High energy": over 30 kWh. "Short-term": sojourn time below 2h; "Medium-term": between 2h and 4h; "Long-term": over 4h.

6.4 Graph Clustering Results

To apply the seven graph clustering algorithms described in Sections 3.4-3.10, we first need to fine-tune their similarity threshold. Using grid search in [0, 1] with a step of 0.01, we measure the number of clusters each algorithm generates as well as the corresponding entropy of the cluster sizes. The goal is to configure each algorithm to generate clusters satisfying two requirements:

1. Their number is low, so that they can be easily interpreted.
2. Their size is balanced, avoiding situations where a single cluster involves the vast majority of records, a situation that may yield high scores for all three evaluation measures, but provides no insights into the behavioral patterns of EV users.

The results with respect to these requirements are presented below, in *Figure 17* and in *Figure 18*, respectively. Note that we had to exclude Cut and Markov Clustering from our analysis, due to their excessively high space and time complexity, respectively. The former requires more main memory than the available one (64 GB), while the latter did not terminate within 24 hours, regardless of the similarity threshold.

Starting with *Figure 17*, we observe that all algorithms exhibit insignificant variation in the number of clusters for all thresholds up to 0.90. The reason is that after normalization, the three features selected in Section 4.4 yield very high similarities between most pairs of records. As a result, most thresholds below 0.90 prune a negligible number of edges from the similarity graph. In fact, Connected Components clustering places all records in the same, single cluster for thresholds up to 0.95, while Correlation Clustering generates a single clustering regardless of the similarity threshold. In contrast, the more elaborate processing of Center and Ricochet SR Clustering enables them to generate a very high number of clusters, regardless of the similarity threshold (note the log scale of the vertical axis in *Figure 17*). The two algorithms converge to ~100 clusters for the highest similarity thresholds, whereas Connected Components and Merge-Center Clustering converge to ~10 clusters.

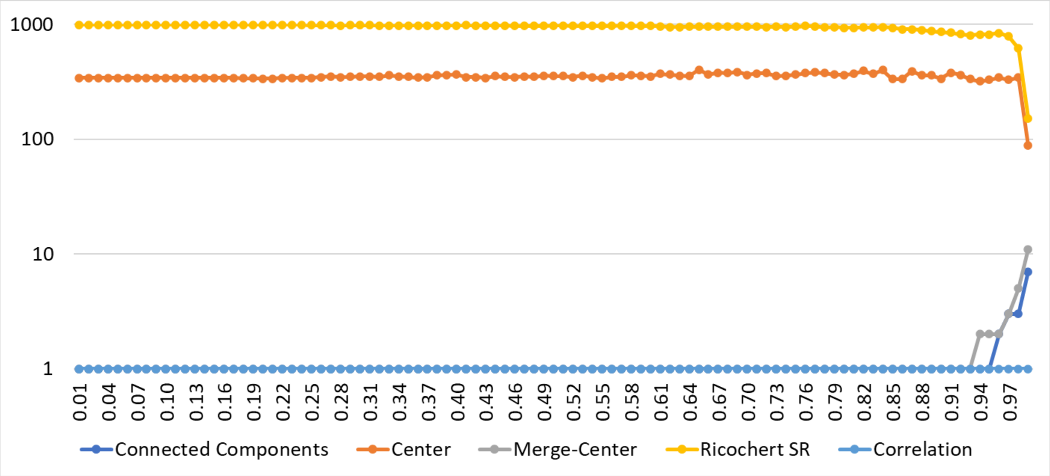


Figure 17. Number of clusters per similarity threshold and clustering algorithm.

To assess the usefulness of the resulting clusters, we estimate the entropy of their sizes across all similarity thresholds. Note that cases with a single cluster correspond to zero entropy. We observe that Connected Components and Merge-Center clustering exhibit the highest entropy with the highest similarity threshold (0.99), for which they also yield the maximum number of clusters. However, the actual value of entropy is practically zero, which indicates that one of the clusters dominates all others, comprising almost all records. Thus, the usefulness of the corresponding clusters is limited.

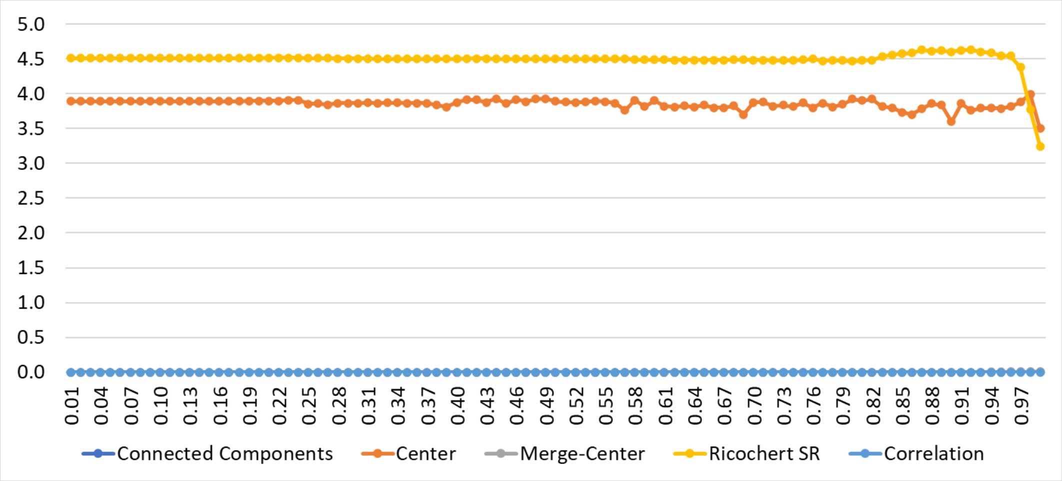


Figure 18. Entropy of cluster sizes per similarity threshold and clustering algorithm.

For Center and Ricochet SR Clustering, there are insignificant variations in entropy across all thresholds, except the largest ones, where both algorithms exhibit a steep decrease. The reason is that in every case, the maximum value of entropy is log|C|, where |C| denotes the corresponding number of clusters. As a result, their entropy is bounded to much lower values for the largest similarity thresholds. Entropy is maximized for t=0.98 and t=0.87 for Center and Ricochet SR Clustering, respectively.

These results are summarized in Table 5. Note that for all algorithms are represented by two different configurations, except Connected Components. For Center and Ricochet SR Clustering, we considered both the threshold maximizing the entropy of cluster size and the threshold minimizing the resulting clusters. For Merge-Center Clustering, we report the similarity threshold that maximizes both the entropy and the number of clusters (0.99), and the threshold that maximizes the evaluation measures (0.94).

Table 5. Summary of graph clustering performance. The best performance is highlighted in bold.

Method	Connected Components	Center1	Center2	Merge-Center1	Merge-Center2	Ricochert SR1	Ricochert SR2
Number of clusters	7	345	88	11	2	896	152
Maximum cluster size	21,794	1,448	1,375	21,781	21,799	3,508	3,715
Portion of entities in largest cluster	99.97%	6.64%	6.31%	99.91%	99.99%	16.09%	17.04%
Similarity threshold	0.99	0.98	0.99	0.99	0.94	0.88	0.99
Entropy	0.003	3.999	3.502	0.009	0.001	4.633	3.248
Silhouette Coefficient	-0.127	-0.571	-0.1479	-0.184	**0.325**	-0.575	-0.235
Davies-Bouldin Index	0.681	12.064	4.584	0.829	**0.634**	16.282	7.258
Calinski-Harabasz Index	1.96	189.81	696.86	3.23	**4.26**	75.26	442.31

The outcomes of graph clustering algorithms can be distinguished into two types:

1. A large number of clusters, whose sizes are balanced, but their performance with respect to the three evaluation measures is quite low: the Silhouette Coefficient is negative, the Davies-Bouldin Index is very high and the Calinski-Harabasz Index is significantly lower than that of K-Means, GMM and Hierarchical Clustering. This performance, which indicates strong dissimilarities between the records inside each cluster, applies to all configurations of Center and Ricochet SR Clustering. Regardless of the evaluation measures, the excessively large number of generated clusters has the additional disadvantage of hampering the qualitative analysis of these clusters.
2. A few clusters with highly imbalanced sizes, yielding low entropy scores close to 0. This pattern applies to Connected Component and Merge-Center Clustering, with their evaluation measures exhibiting very high scores. In fact, the Davies-Bouldin Index is consistently lower than that of K-Means, GMM and Hierarchical clustering to a significant extent, thus indicating more homogeneous clusters. Moreover, the Silhouette Coefficient of Merge-Center Clustering with t=0.94 is practically identical with the highest scores achieved so far, i.e., those of K-Means and Hierarchical Clustering. Only the Calinski-Harabasz Index is significantly lower than all other algorithms, including the other graph clustering techniques. In theory, these results are quite positive, but in practice they lack any usefulness. The reason is that in all three cases, a single cluster is essentially created, as the largest one contains more than 99% of all records.

These results indicate that the performance of graph clustering algorithms is inferior to K-Means, GMM and Hierarchical clustering, providing no useful insights into EV charging patterns.

6.5 Summary of Results

Unlike the graph clustering algorithms, K-means and GMM delivered consistent and effective results for identifying meaningful EV charging profiles with practical applications. The K-means method produced the highest overall scores, while GMM yielded more specific and extreme profiles, which can be particularly interesting for analyzing the most diverse sessions. Hierarchical clustering it achieved high values in all evaluation scores, but these did not translate into better qualitative profiles, as they are more generic, overlapped, and less visually defined clusters. Table 6 summarizes these metrics and parameters selected for the best performing clustering algorithms, with the best evaluation scores highlighted in bold.

Table 6. Summary of the performance of representative-based and hierarchical clustering algorithms.

	K-means	**GMM**	**Hierarchical**
Best number of clusters	10	8	7
Parameters	-	Tied Covariance	Ward’s Method
Elbow Method	$k = \{7,8,9,10\}$	-	-
Silhouette Coefficient	**0.326**	0.309	0.322
Davies-Bouldin Index	**0.983**	1.061	1.036
Calinski-Harabasz Idex	10,715.45	9,259.41	**11,777.57**

7 Conclusions

The EV charging profiles provide information about the times of day when more or fewer charging sessions occur, whether the sessions are of high or low energy and with high or low flexibility potential. To identify these patterns, we applied the following methodology:

1. First, pre-processing cleaned the original data from missing and outlier values.
2. Feature engineering defined new features and selected the most essential ones. The resulting dataset comprises 21,801 records involving 3,184 different users and 313 different charging points, with each record corresponding to a different charging session, as described by three complementary, non-redundant features: start datetime, volume (kWh delivered) and sojourn time. Note that these records predominantly contain quick-stay sessions, due to the EVSE locations in publicly available infrastructures.
3. Finally, a wide variety of clustering algorithms was applied, covering all major types, from representative-based to hierarchical and graph clustering.

Among the tested techniques, the graph clustering algorithms exhibit rather low performance, due to the low number of features, which after normalization yield very high similarities between most records. As a result, the similarity thresholds we considered in [0,1] with a step of 0.01 yield a single cluster or very few, quite imbalanced ones in combination with most graph clustering algorithms. More fine-grained similarity thresholds (e.g., in [0.9990, 0.9999] with a step of 0.0001) will probably produce better results. However, this means that graph clustering is excessively sensitive to the similarity threshold, rendering its fine-tuning an non-trivial task.

In these similarity settings, configuring the number of final clusters is more straightforward than fine-tuning the similarity threshold. As a result, the representative-based and hierarchical clustering algorithms outperform the graph one to a significant extent. Among them, K-means performed better than GMM and Hierarchical Clustering, with its best performance corresponding to K=10. This outcome was achieved after an extensive study on the number of clusters that provided the optimal balance between the three considered evaluation measures (Silhouette, Davies-Bouldin, and Calinski-Harabasz index) and the usefulness of the resulting profiles. The configurations of the best scores often led to meaningless typical profiles, requiring a more in-depth analysis. Selecting the ideal covariance type for GMM clustering and distance measure for Hierarchical clustering was also crucial; tied covariance and Ward's Method were consistently the most appropriate options, respectively.

On the downside, K-means and GMM are sensitive to the random initialization of their algorithms (as expressed through the seed that is responsible for these initializations in the scikit-learn library, defined through the parameter random state). As a result, the seed affects the results and their reproducibility. To address this issue, we conducted multiple analyses to determine the optimal random state for each study.

The results of this deliverable intend to help Utilities, Distribution System Operators (DSOs), and CPOs to perform a successful and intelligent integration of EVs into the energy system, providing them with valuable information about the charging behavior of EVs and users. They can also be helpful in future activities related to power systems planning and in the coordination of EVs with Renewable Energy Sources (RES).

References

[1] K. Dimitriadou, N. Rigogiannis, S. Fountoukidis, F. Kotarela, A. Kyritsis, and N. Papanikolaou, “Current Trends in Electric Vehicle Charging Infrastructure; Opportunities and Challenges in Wireless Charging Integration,” Energies, vol. 16, no. 4, p. 2057, Feb. 2023. [Online]. Available: https://www.mdpi.com/1996-1073/16/4/2057

[2] J. R. Helmus, M. H. Lees, and R. van den Hoed, “A data driven typology of electric vehicle user types and charging sessions,” Transportation Research Part C: Emerging Technologies, vol. 115, p. 102637, Jun. 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0968090X19315414

[3] A. Martz, U. Langenmayr, S. Ried, K. Seddig, and P. Jochem, “Charging Behavior of Electric ¨ Vehicles: Temporal Clustering Based on Real-World Data,” Energies, vol. 15, no. 18, p. 6575, Sep. 2022. [Online]. Available: https://www.mdpi.com/1996-1073/15/18/6575

[4] S. Shahriar and A. R. Al-Ali, “Impacts of COVID-19 on Electric Vehicle Charging Behavior: Data Analytics, Visualization, and Clustering,” Applied System Innovation, vol. 5, no. 1, p. 12, Jan. 2022. [Online]. Available: https://www.mdpi.com/2571-5577/5/1/12

[5] Y. Shen, W. Fang, F. Ye, and M. Kadoch, “EV Charging Behavior Analysis Using Hybrid Intelligence for 5G Smart Grid,” Electronics, vol. 9, no. 1, p. 80, Jan. 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/1/80

[6] Y. Xiong, B. Wang, C.-C. Chu, and R. Gadh, “Electric Vehicle Driver Clustering using Statistical Model and Machine Learning,” in 2018 IEEE Power & Energy Society General Meeting (PESGM). Portland, OR: IEEE, Aug. 2018, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/document/8586132/

[7] G. Van Kriekinge, C. De Cauwer, N. Sapountzoglou, T. Coosemans, and M. Messagie, “Electric Vehicle Charging Sessions Generator Based on Clustered Driver Behaviors,” World Electric Vehicle Journal, vol. 14, no. 2, p. 37, Feb. 2023. [Online]. Available: https://www.mdpi.com/2032-6653/14/2/37

[8] A. Gerossier, R. Girard, and G. Kariniotakis, “Modeling and Forecasting Electric Vehicle Consumption Profiles,” Energies, vol. 12, no. 7, p. 1341, Apr. 2019. [Online]. Available: https://www.mdpi.com/1996-1073/12/7/1341

[9] L. M. L. Cam and J. Neyman, Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability: Weather modification. University of California, 1967, google-Books-ID: IC4Ku 7dBFUC.

[10] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum Likelihood from Incomplete Data via the EM Algorithm,” Journal of the Royal Statistical Society. Series B (Methodological), vol. 39, no. 1, pp. 1–38, 1977. [Online]. Available: http://www.jstor.org/stable/2984875

[11] M. Steinbach, G. Karypis, and V. Kumar, “A Comparison of Document Clustering Techniques,” University of Minnesota Digital Conservancy, Report, May 2000. [Online]. Available: http://conservancy.umn.edu/handle/11299/215421

[12] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial Databases with Noise,” Knowledge Discovery and Data Mining, pp. 226–231, Jan. 1996. [Online]. Available: https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf

[13] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, “OPTICS: ordering points to identify the clustering structure,” in Proceedings of the 1999 ACM SIGMOD international conference on Management of data. Philadelphia Pennsylvania USA: ACM, Jun. 1999, pp. 49–60. [Online]. Available: https://dl.acm.org/doi/10.1145/304182.304187

[14] H. Jia, S. Ding, X. Xu, and R. Nie, “The latest research progress on spectral clustering,” Neural Computing and Applications, vol. 24, no. 7-8, pp. 1477–1486, Jun. 2014.

[15] M. J. Zaki and W. Meira, Jr, Data Mining and Machine Learning: Fundamental Concepts and Algorithms, 2nd ed. Cambridge University Press, 2020.

[16] P. H. A. Sneath and R. R. Sokal, Numerical Taxonomy. W H Freeman & Company, Jan. 1973.
[17] K. Florek, J. Łukaszewicz, J. Perkal, H. Steinhaus, and S. Zubrzycki, “Sur la liaison et la division des points d’un ensemble fini,” Colloquium Mathematicum, vol. 2, no. 3-4, pp. 282–285, 1951. [Online]. Available: http://eudml.org/doc/209969
[18] F. J. Rohlf, “12 Single-link clustering algorithms,” Handbook of Statistics, Jan. 1982.
[19] T. Sørenson, A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species Content and Its Application to Analyses of the Vegetation on Danish Commons, ser. Biologiske skrifter. I kommission hos E. Munksgaard, 1948. [Online]. Available: https://www.royalacademy.dk/Publications/High/295_S%C3%B8rensen,%20Thorvald.pdf
[20] R. Sokal, C. Michener, and U. of Kansas, A Statistical Method for Evaluating Systematic Relationships. University of Kansas, 1958. [Online]. Available: https://bit.ly/3LuRirh
[21] J. D. Ward, “Hierarchical Grouping to Optimize an Objective Function,” Journal of the American Statistical Association, vol. 58, no. 301, p. 236, Mar. 1963.
[22] P. B. G. Alvez, “Inference of a human brain fiber bundle atlas from high angular resolution diffusion imaging,” phdthesis, Universite Paris Sud - Paris XI, Oct. 2011. [Online]. Available: https://theses.hal.science/tel-00638766
[23] T. H. Haveliwala, A. Gionis, and P. Indyk. Scalable Techniques for Clustering the Web. In Proc. of the Int’l Workshop on the Web and Databases (WebDB), pages 129–134, Dallas, Texas, USA, 2000.
[24] O. Hassanzadeh and R. J. Miller. Creating Probabilistic Databases from Duplicated Data. Technical Report CSRG-568, University of Toronto, To appear in The VLDB Journal, Accepted on 26 June 2009.
[25] D. T. Wijaya and S. Bressan. Ricochet: A Family of Unconstrained Algorithms for Graph Clustering. In Proc. of the Int’l Conf. on Database Systems for Advanced Applications (DASFAA), pages 153–167, Brisbane, Australia, 2009.
[26] N. Bansal, A. Blum, and S. Chawla. Correlation Clustering. Machine Learning, 56(1-3):89–113, 2004
[27] S. van Dongen. Graph Clustering By Flow Simulation. PhD thesis, University of Utrecht, 2000
[28] G. W. Flake, R. E. Tarjan, and K. Tsioutsiouliklis. Graph Clustering and Minimum Cut Trees. Internet Mathematics, 1(4):385–408, 2004.
[29] P. J. Rousseeuw, “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis,” Journal of Computational and Applied Mathematics, vol. 20, pp. 53–65, Nov. 1987. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/0377042787901257
[30] D. L. Davies and D. W. Bouldin, “A Cluster Separation Measure,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-1, no. 2, pp. 224–227, Apr. 1979. [Online]. Available: http://ieeexplore.ieee.org/document/4766909/
[31] T. Calinski and J. Harabasz, “A dendrite method for cluster analysis,” Communications in Statistics - Theory and Methods, vol. 3, no. 1, pp. 1–27, 1974. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/03610927408827101