UNIVERSITY OF ŽILINA

FACULTY OF MANAGEMENT SCIENCE AND INFORMATICS

DIPLOMA THESIS

MARTIN JANČURA

Regularly updated predictions of electric vehicle connection duration.

Tutor: Ing. Milan Straka, PhD.

Registration number: 1671/2021

Žilina 2022

UNIVERSITY OF ŽILINA

FACULTY OF MANAGEMENT SCIENCE AND INFORMATICS

DIPLOMA THESIS

Field of study: Informatics

MARTIN JANČURA

Regularly updated predictions of electric vehicle connection duration.

University of Žilina

Faculty of Management Science and Informatics

Department of mathematical methods and operations research.

Žilina 2022

ŽILINSKÁ UNIVERZITA V ŽILINE, FAKULTA RIADENIA A INFORMATIKY.

ZADANIE TÉMY DIPLOMOVEJ PRÁCE.

Študijný program: Inteligentné informačné systémy

Meno a priezvisko	**Osobné číslo**
Martin Jančura	559844

Názov práce v slovenskom aj anglickom jazyku

Pravidelne aktualizované predikcie doby pripojenia elektrických vozidiel

Regularly updated predictions of electric vehicle connection duration

Zadanie úlohy, ciele, pokyny pre vypracovanie (Ak je málo miesta, použite opačnú stranu)

Cieľ diplomovej práce:
Cieľom práce je vytvoriť predikčné modely, ktoré sú schopné predikovať dobu pripojenia v pravidelných intervaloch, na základe dát z nabíjania elektrických vozidiel.

Obsah:
V literatúre sú pre aplikácie v elektromobilite väčšinou dostupné modely, ktoré poskytujú predpoveď jeden krát pre jedno pozorovanie. Naším cieľom je preto vytvoriť model, ktorý je schopný pravidelne obnovovať predpovede na základe prichádzajúcich dát a vylepšiť tak celkovú presnosť predpovedí. Dôležitou súčasťou práce je takisto extrahovať z dát vstupné prediktory pre model. Pravidelne aktualizované predpovede slúžia najmä ako vstup pre algoritmy inteligentného nabíjania, ktorých jednoduché verzie môžu byť použité na overenie riešenia. Samotná práca bude prebiehať nasledovne:
1. Analýza súčasného stavu na základe dostupnej literatúry,
2. návrh riešenia,
3. spracovanie dostupných dát z nabíjania elektrických vozidiel a tvorba prediktorov pre model,
4. implementácia a trénovanie modelu,
5. vyhodnotenie výsledkov.

Meno a pracovisko vedúceho DP: Ing. Milan Straka, PhD., KMMOA, ŽU
Meno a pracovisko tútora DP:

8.2.2022

vedúci katedry
(dátum a podpis

- 3 MAR. 2022

Zadanie zaregistrované dňa ..... pod číslom 1671/2021 podpis _____

Prehlásenie

Čestne vyhlasujem, že som prácu vypracoval samostatne s využitím dostupnej literatúry a vlastných vedomostí. Všetky zdroje použité v bakalárskej práci som uviedol v súlade s predpismi.

V Žiline, dňa 20.4.2022 Martin Jančura

Poďakovanie

Touto cestou vyslovujem poďakovanie pánom Ing. Milan Straka, PhD a prof. Ing. Ľuboš Buzna, PhD. za pomoc, odborné vedenie a cenné rady pri výskume, ktorý bol základom tejto dimplomovej práce. Taktiež spoločnosti ElaadNL za poskytnutie kľúčových a nenahraditeľných dát, bez ktorých by práca nemohla vzniknúť.

Abstrakt

JANČURA MARTIN: *Pravidelne aktualizované predikcie doby pripojenia elektrických vozidiel* [Diplomová práca]

Žilinská Univerzita v Žiline, Fakulta riadenia a informatiky, Katedra matematických metód a operačnej analýzy.

Vedúci: Ing. Milan Straka, PhD.

Stupeň odbornej kvalifikácie: Inžinier v odbore Informatika v Žiline.

FRI ŽU v Žiline, 2022 — 76 s.

Elektrické vozidlá sú sľubným riešením na zníženie rastúcich emisií CO_2 z dopravy za predpokladu, že potrebná elektrina bude pochádzať z obnoviteľných zdrojov energie. Náhodnosť obnoviteľných zdrojov a dopytu po nabíjaní elektrických vozidiel vyžaduje inteligentné schémy nabíjania. Inteligentné nabíjanie dosahuje lepšie výsledky, ak má prístup k presnejším predpovediam správania sa nabíjania. Z tohto dôvodu by mohlo byť prínosné inteligentné nabíjanie, ktoré dynamicky aktualizuje plán nabíjania. V tejto práci skúmame potenciál zlepšenia presnosti predpovede dĺžky pripojenia elektrického vozidla pomocou aktualizácie predpovede počas pripojenia elektrického vozidla k nabíjacej stanici. Porovnávame jeden model a viacnásobný model pre aktualizáciu predpovede pravidelne a nepravidelne distribuovanú v čase. Viacnásobný model dosahuje najlepšie výsledky, pričom zlepšuje presnosť predpovede až do 40 % v porovnaní s konvenčnými prístupmi. Vyššia frekvencia aktualizácie tesne po pripojení je efektívnym riešením na zvýšenie presnosti predpovede. Neskôr postačuje pravidelná aktualizácia. Vzhľadom na inteligentné nabíjanie, ktoré pravidelne aktualizuje plán nabíjania, dosahuje najlepšie výsledky viacnásobný model s nepravidelne distribuovanými časmi aktualizácie, pričom podstatne znižuje spotrebu energie v špičke, a zároveň výrazne nezvyšuje nenabitú požadovanú energiu.

Kľúčové slová: elektrické vozidlá, inteligentné nabíjanie, aktualizované predpovede, strojové učenie, dátová veda.

Abstract

JANČURA MARTIN: *Regularly updated predictions of electric vehicle connection duration.* [Diploma thesis]

University of Žilina, Faculty of Management Science and Informatics, Department of mathematical methods and operations research.

Tutor: Ing. Milan Straka, PhD.

Qualification level: Engineer in field Informatics in Žilina:

FRI ŽU v Žiline, 2022 — 76 p.

Electric vehicles are promising to alleviate the increasing CO_2 emissions from transport, provided that renewable energy sources generate the demanded electricity. The stochasticity of renewable energy sources and charging demand require intelligent charging schemes. Smart charging achieves better performance when it is driven by reasonably accurate predictions of charging behaviour. Hence, for smart charging that dynamically updates a charging schedule, updating the predictions of charging behaviour could be beneficial. In this paper, we explore the potential to improve the accuracy of prediction models of the connection time to a charging station by updating the predictions as the charging sessions unfold. We compare a single model with multiple models for regularly and irregularly spaced updates in time. The multiple models with irregular updates achieve the best performance while improving the prediction accuracy up to 40 %, compared to conventional approaches. It is efficient to update the predictions with higher frequency in the very early stages of charging sessions. Later on, regular updates are sufficient. Moreover, multiple model with updates irregularly spaced in time performs best, considering the smart charging that dynamically updates a charging schedule, as it importantly mitigates the energy charged in peak periods and does not significantly increase demanded but not charged energy.

Keywords: electric vehicles, smart charging, updated predictions, machine learning, data science.

Contents

List of Figures

List of Tables

List of Abbreviations

EV	Electric vehicle
KDE	Kernel density estimation
ToU	Time-of-use
BaU	Business-as-usual
LGBM	LightGBM algorithm
GBRT	Gradient Boosted Regression Trees
TPE	Tree-structured Parzen Estimator Approach
TP	True positive
TN	True negative
FP	False positive
FN	False negative

Chapter 1

Introduction

Recently, the humanity seeks options to decrease anthropogenic CO_2 emissions, as they are correlated with global warming [1]. One of the promising solutions to decrease CO_2 emissions are electric vehicles (EVs), when charged from renewable energy sources. The stochastic nature of the EV charging demand and renewable energy sources asks for a smart charging to coordinate the charging process as a part of demand response approach. Efficient smart charging requires estimates of the future developments, e.g., predictions of the charging behaviour. In the literature, several prediction approaches already exist. However, the predictions are mostly done at the arrival of an EV to a charging station and remain valid for the whole duration of a charging session. For smart charging schemes that dynamically update a charging schedule, it would be possible to update also predictions of charging behaviour. Hence, there is potential to reach higher efficiency of dynamic smart charging, if more accurate predictions of charging behaviour can be achieved by updates. Moreover, demand for providing the information about the charging point availability has been emerged, as the limited number of public chargers has become one of the major impediment for spreading the EVs [2]. The accurate prediction of EV connection duration may be beneficial in order to announce EV drivers about future availability.

From upper mentioned reasons, this paper handles prediction of the EV connection duration using novel update perception. In this chapter, state-of-art prediction approaches of EV charging behaviour as well as methods coping with updating predictions are inspected. Consequently, main contribution realms are described. Later on, we denote the terminology and

write up the list of used symbols for lucidity. The first section of Chapter 2 introduces a dataset on which the experiments are performed. Additionally, we claim some expectations and assumptions along describing main characteristics of the dataset in order to propose appropriate features. Since feature processing can be computationally expensive, in the last section of Chapter 2, algorithms and implementation are closely described. Chapter 3 represents the major point of this paper, as it describes update strategies, metrics and application for smart charging and charging point availability. Result are shown and analysed in Chapter 4. In the conclusion, all goals and results are summed up as well as future outlooks are described.

1.1 Literature review

We divide literature into two fields according to the thesis goals. Firstly, we explore the current studies coping predictions of EV connection duration. Later on, we explore the methods and studies handling updates in related fields, since in this thesis we try to improve predictions by updating.

1.1.1 Predictions of EV connection durations.

Among the most popular forecasting techniques in the EV field are the forecast of aggregated demand [3]. However, for smart charging algorithms, individual forecasts can allow a better control of individual EVs. The individual predictions arose with public availability of charging datasets. Several solutions are providing estimations of charging session attributes.

In [4] authors predicted the energy consumption and session duration using data from past charging sessions. The prediction is made in two steps - the prediction of the connection duration is used as a feature for the prediction of the energy consumption. In [5] authors predicted connection duration of EVs, based on past session data. Since users behave in some generic charging patterns, in [6] user's behaviour is predicted using clustering technique. The main contribution of the article is to highlight the advantages for the prediction in smaller groups containing sessions with similar charging pattern instead of the prediction on the whole comprehensive dataset, especially for the predictions of EV connection duration. Clustering engenders more effective learning in smaller groups, confirmed by results published in the

article - the higher cluster count is, the better accuracy is achieved. Additionally, some sensible patterns of users' behaviour were identified. Moreover, charging patterns are used also in [7], where seasonality and periodicity of the users' behaviour are captured in time series predicting first daily unplugging. Another strategy to predict duration available for smart charging is to estimate the time the EV remains parked after charging (called idle time), as in [8], where three regression algorithms were used.

Although these approaches improve the accuracy compared to benchmark models, none of them considered updating the predictions in time. Such updates have the potential of improving the predictions.

1.1.2 Prediction update

Incremental learning is the main field handling continuous usage of data stream to extend the knowledge of a model. This brings the improvement of the prediction accuracy. It is widely used for model rebuilding and online or large data stream processing.

Authors in [9] applied incremental learning using adaptive boosting for classification with goal to accumulate and transform the information from online data stream. In [10] authors realise incremental learning using support vector machines, utilising that support vector machines algorithm reduces data to support vectors, while comparing it to conventional whole dataset approach. Prediction of energy consumption in particular day hours is proposed in [11] using neural networks and mini batch learning to retrain and adapt model to seasonal weather conditions and daily trends. Online AdaBoost and online bagging algorithms are used for online learning in [12]. Novel AdaBoost method for online learning was proposed in [13] for real-time learning, complex background systems learning, visual tracking and image recognition.

Update of predictions can be beneficial for wide scale of duration predictions, such as duration of system outages [14] or incident duration [15]. In [15] authors justify the use of the incremental with the fact, that not all information is available at the beginning of the incident and a lot of information arrives during the incident. Similarly, in [14] authors use incremental approach, as new information arise during power outage prediction, possibly influencing

the duration of the outage. Many applications of updated predictions are applied to public transport. Prediction of delays [16, 17], where in both cases Bayesian network is used for predicting the train delays. In [18], train delays are predicted using timed event graph with dynamic arc weights.

In the context of individual charging behaviour, we have not identified any paper exploring the updates of predictions.

1.2 Goals

In the following, we describe the main goals and contributions of the thesis, which can be divided into tree separate groups:

1. To emphasise the benefit of updating.
2. Direct applications for EVs, e.g., smart charging and charging point availability.
3. To compile the updating methodology, possibly useful for related field.

Since none of current studies among the most popular forecasting techniques in the EV field updates predictions, in this paper we examine the impact of updating the predictions. Therefore, we identify the scientific contribution to open new perception in this realm, possibly engendering novel techniques and studies in the EV's field or related fields. Besides that, improved predictions might be used also for smart charging and real-time charging point availability. The proposed methodology is widely usable for prediction of the generic process duration.

1.2.1 Smart charging

The ability to minimise peak demand and network congestion allowing usage of cheaper, low carbon generation, is the key feature of a smarter energy system. To reach zero road emission, some countries, such as the United Kingdom [19], manage the impact of EVs on the electricity system using smart charging.

Smart charging algorithms can be classified into three categories [20]: smart grid oriented, aggregator oriented and driver-oriented. Review of smart charging algorithms [21] highlights how possible future directions the development of prediction methods that can deal with uncertainty in driver mobility behaviour and the development of approaches properly balancing prediction accuracy, model simplicity, and data requirements. There are many optimisation objectives that the smart charging follows, such as minimisation of charging cost and peak power minimisation [22, 23, 24, 25].

The EV charging flexibility refers to charging coordination in the grid [26]. Two strategies, the peak flattening and the maximisation of renewable energy usage, are analysed in [26] from the perspective of the EV charging flexibility. Two scenarios' benefit were estimated in [24], where authors considered non-residential smart charging. In the first scenario, behind-the-meter EV aggregations are combined with the time-of-use (ToU) smart charging scheme, resulting in significant monetary savings. The minimisation of the peak load engendering peak power decrease in the second scenario.

Another approach to EV charging flexibility concerns the willingness of EV drivers to change their charging habits. In [27], the authors concerned the willingness of EV drivers to change their charging habits. In the experiment, EV drivers used a mobile phone app to cancel or change the speed of charging. Results confirmed drivers adaptation to the system requirements. However in a few weeks, the usage of this mobile app feature decreased to 2-3% of all sessions.

Zweistra et al. [28] performed a smart charging experiment to decrease the peak load. In [28], the authors evaluated the number of sessions that were terminated before the charging was completed, in order to decrease the peak load. They confirmed the potential contribution of smart charging, since the number of sessions terminated before the charging was completed was low. Obviously, the acceptance of smart charging can be problematic regarding negative impact on the users' mobility [29].

1.2.2 Real-time charging point availability

Nowadays, a lack of chargers could stall the electric-vehicle revolution [30]. Moreover, the limited count of public charging stations represents the major obstacles for widespread EV adoption. Thus, the development of the management for charging point is necessary. For instance, Google will now show EV drivers real-time availability of charging stations directly inside Google Maps [31].

Additionally, the charging point availability can be provided in advance. Some current papers try to handle prediction of the charging point availability from macroscopic perception, as in [32]. The occupation status of charging infrastructure or charging point availability for the next day is considered in [33, 34]. Since drivers plan the sessions operatively, the critical point for prediction is close to the end of the currently active sessions. Hence, some studies aim to forecast electric charging station availability in the near future, such as in [35].

In general, operative information about the current and future availability of the charging points for drivers, can be provided by systems using predictions of the EV connection duration. Especially, update approach can bring the improvement of the predictions exactly in the right time.

1.2.3 Potential usage of the methodology

To emphasise the contribution of this work, in this subsection the potential methodology usage is described. Since we have not identified any paper exploring the updates of predictions in the context of individual charging behaviour, we propose novel methodology useful as well for different fields handling prediction of the process duration, i.e. process finish.

Such prediction of the duration is important component for health sector. The improvement of the accuracy of the prediction using update can be beneficial for both, patient and hospital management. Prediction of surgery duration, as is predicted in [36], can improve the surgery scheduling. Nowadays, predicting the hospital length of stay for patients with COVID-19 infection is essential [37].

Some fields already use the prediction update. Therefore, novel methodology can improve current state-of-art methods, such as in [14] for prediction of the system outages duration or

for incident duration of papers mentioned in [15].

Generally, we suppose many other applications and usage of our novel methodology in related predictions coping with process durations.

1.3 Terminology

For the paper's needs, we follow terminology stated in [38] and used in [39], where a connector is defined as a physical interface between an EV and charging infrastructure through which electricity is delivered. A charging point is an energy delivery device consists of one or more connectors, due to many connectors types exist. Despite that, only one connector can be active at a time at the charging point. A charging capacity determines maximal charging power, given in kW, for the charging point. The composition of one or more charging points is labelled as a charging station, where all charging points are attached to the same charging station for the ambient environment.

Every charging session k starts with plugging the EV and is determined by arrival of an EV t_k^{arr} and departure time of an EV t_k^{dep}. Time difference in between t_k^{arr} and t_k^{dep} is called connection duration (in our scenario, response variable y_k), composed of a charging duration $t_k^{char} - t_k^{arr}$ and an idle time. The graphical illustration of the session is visualized in Figure 1.1.

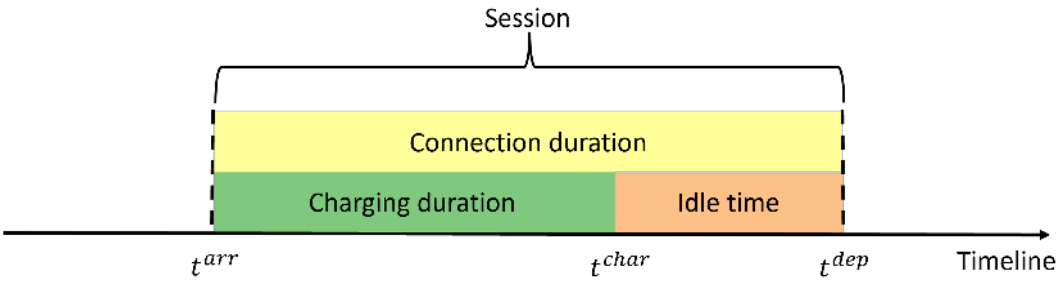

Figure 1.1: Graphical illustration of the session.

1.3.1 List of symbols

For charging session, we denote the following symbols:

- t_k^{arr} - observed arrival time of a vehicle k to a charging station,
- t_k^{dep} - observed departure time of a vehicle k from a charging station,
- t_k^{char} - time when a charging of the session k was terminated,
- E^n - demanded but not delivered energy,
- E^p - the energy charged in peak period for peak price,
- E^o - the less expensive energy charged in offpeak periods.

Moreover, we use some specific terms and symbols to describe the update of the prediction. Hence, we write up them in the following list:

- N - number of charging sessions,
- M - number of updates of a prediction,
- M_k - number of updates of a prediction of the k-th session,
- $\boldsymbol{x}$ - vector of feature values associated with one observation,
- $\boldsymbol{x}_k$ - vector of feature values associated with k-th observation (session),
- t_k^i - time offset of the i-th prediction update since the start of k-th session,
- t^1 - time offset of the first prediction,
- y_k - response variable (connection duration),
- $\hat{y}_k^i$ - estimate of the connection duration of k-th charging session resulting from the prediction update made at the time offset t_k^i since the session start.
- δ - length of the time interval between two prediction updates,
- δ_k^i - length of the time interval between i-th and $i+1$-th connection duration prediction updates of the session k,

Chapter 2

Data and features

In the first section of this chapter, we introduce the dataset on which we perform our experiments to consider novel proposed features and methodology arising from the updating approach. Some expectations and assumptions are also claimed, as we try to describe main characteristics of the dataset itself. In the second section, we use the knowledge from the exploration of the dataset and try to propose appropriate features. Since feature processing can be computationally expensive, in the last section, algorithms and implementation are closely described.

2.1 EVnetNL dataset

We performed our experiments on the EVnetNL dataset collected by the knowledge and innovation centre in the field of smart charging and the charging infrastructure ElaadNL [40]. The dataset comprises two tables, “Transactions" and “Meterreadings". Each charging session in the table “Transactions" is described by charging point and connector identifiers, geographical coordinates, timestamps of initiation and termination, and identifiers of the user RFID cards used to initiate and terminate charging sessions. Table “Meterreadings" describes the charging energy consumption with a 15-minute frequency. The subset of data we used spans from 01/2016 to 07/2018, covers 1731 public and semi-public charging stations (located as in Figure 2.1), about 65k EV drivers, more than 936k charging sessions, and more than 30M meter readings.

Figure 2.1: The location of the charging stations for EVnetNL dataset.

2.1.1 Dataset analysis

Early on, we made a data cleaning. As a part of data cleaning, we found a few sessions with exceedingly large connection duration, sometimes several days or even weeks. Regarding smart charging, such long duration is not of high relevance and for this reason, we capped the connection duration to 24 hours. The distribution of connection duration is shown in Figure 2.2. The majority of connections take less than five hours. Only a minority of sessions takes more than 20 hours.

As we mentioned in literature review, charging sessions indicate some main charging patters in general [6, 7]. Recent studies explored the charging patters similar for both, users and charging stations. In [41], five distinct clusters of daily plug-in EVs charging profiles (corresponding to behaviour of the users) were observed at the public charging stations using pattern analysis.

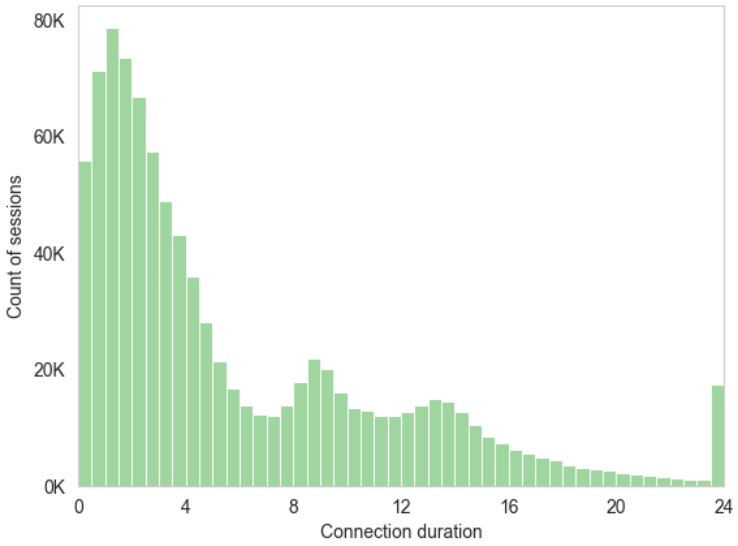

Figure 2.2: Histogram of the connection duration for charging sessions in the EVnetNL dataset.

Some studies already analyzed EVnetNL dataset. Both papers [42, 43] create clusters of charging points, based on parameters of the charging sessions or usage of the charging stations. In [42] four parameters describing popularity, utilization and temporal usage of charging stations. Four groups of stations characterized by distinct usage patterns were identified, resulting in clusters of stations that are not identical. Based on the EV arrival times and the duration of EV connection to the charging station, in [43] authors identify charging matrices belonging to each of 10 clusters identified by hierarchical clustering.

In our scenario, we follow related studies and explore similarities of users and stations separately. According to paper topic, we additionally use connection duration for aggregation in order to find similar charging patterns for diverse groups of sessions. Figure 2.3 displays kernel density estimation [44] (KDE) for connection duration. We explore the impact of the current duration of the sessions on KDE. Panel A shows all sessions from the 2016. Note, that KDE of the connection duration is similar to the histogram in Figure 2.2. Just session with connection duration longer than 3 hours are shown in Figure 2.3B. Similarly, we use sessions lasting more than 5 hours in Figure 2.3C and sessions last more than 10 hours in Figure 2.3D. Figures 2.3B- 2.3D refer to situations, when the current duration of the sessions is equal to

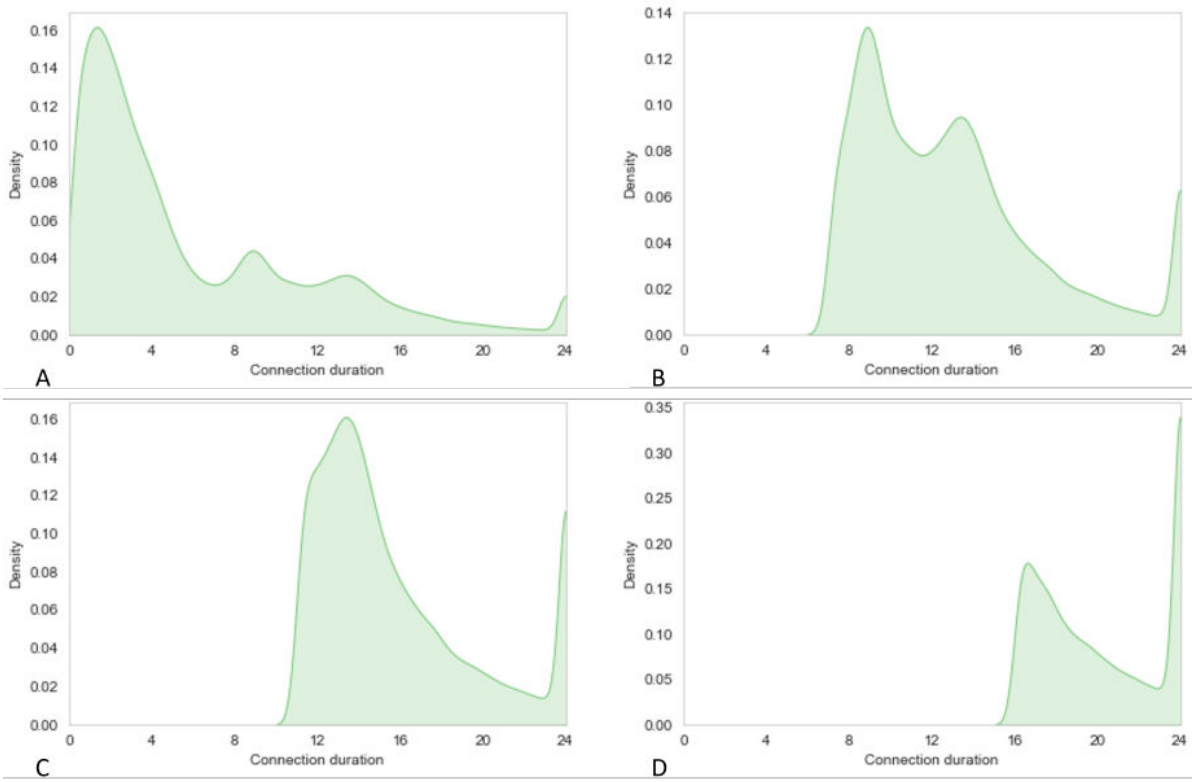

Figure 2.3: Kernel density estimation of the connection duration for different sets of the sessions. Panel A refers to the time offset equals to 0 (all sessions are included). Panel B-D include just the sessions with connection duration grater than 7, 11 and 16 hours.

a particular time offset, and we predict the connection duration of the appropriate sessions. Evidently, the connection duration of the sessions is more predictable, as the current duration is raising.

Since standard deviation of the connection duration (target variable) indicates difficulty of the prediction, we analyse the standard deviation in various time offsets, i.e. different sets of active sessions. Thus, all panels in Figure 2.4 show mean standard deviation in various time offsets. In each time offset, we group sessions by user identification or charging point and calculate standard deviation for each group separately. Hence, just sessions lasting more than time offset determine the standard deviation for each group. As a result, we calculate the mean value from standard deviations of all groups for each time offset t^i. Besides that, we

also plot standard deviation from all active sessions without any aggregation. Panel B takes just sessions of relevant users into consideration. In our scenario, the relevant user has got more than 50 sessions in the dataset.

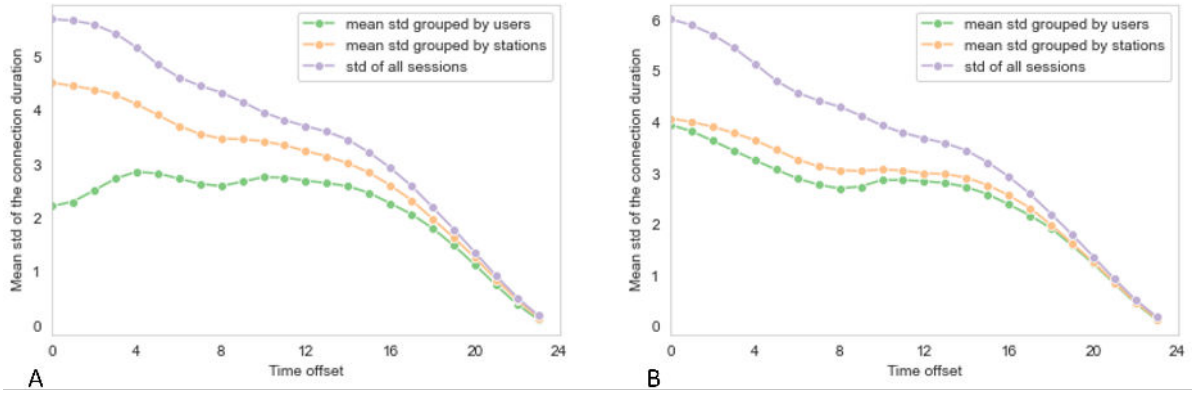

Figure 2.4: Mean standard deviation from active session in time offsets, grouped by users and charging points and not grouped by. Values in panel A are figured out from all sessions. Panel B shows values for users with 50 and more sessions.

Standard deviations are smaller for grouped sessions than for not aggregated sessions. Furthermore, groups of users' sessions reach lower mean standard deviation compared to groups of charging points' sessions. Therefore, the features grouped by users' identification are the most promising for prediction of the EV connection duration. Groups of sessions for charging points also improve the overall standard deviation, thus features aggregation by charging points would be beneficial as well.

The contribution of the update of the predictions is clearly visible in Figure 2.4B, since the standard deviation decreases as the sessions unfold. Similarly, in Figure 2.4A standard deviation decreases with raising time offset for ungrouped sessions and sessions grouped by charging point. Surprisingly, sessions of users with less than 50 session decrease the standard deviation for small values of time offset. Since similar trend is not visible also for mean standard deviation of sessions grouped by charging point and ungrouped sessions, we suspect many occasional users with small count of sessions in the observed period. In the average,

these sessions last less than 5 hours. Additionally, we assume that the majority of the sessions lasting less than 5 hours are sessions of the occasional users.

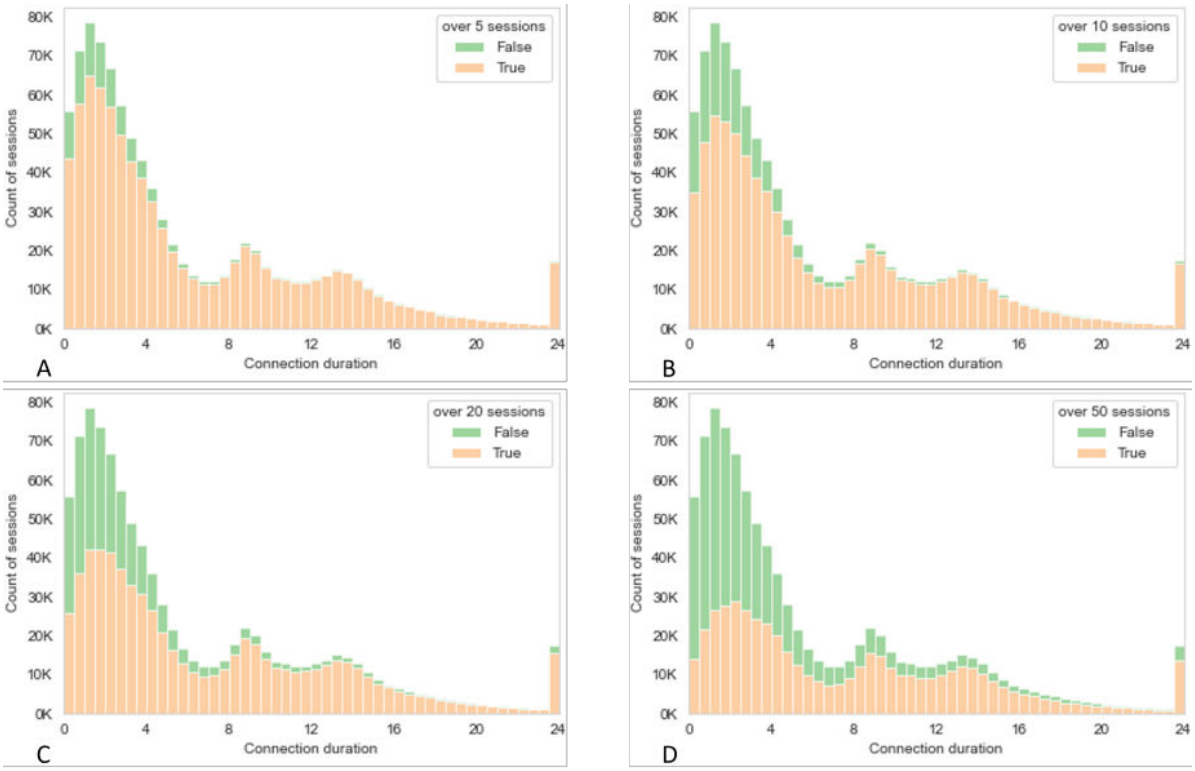

Figure 2.5: Histogram of the connection duration for charging sessions for two separate groups of users. Users are divided base on their session count in observed period. Panels show groups divided by count of session 5, 10, 20 and 50.

All panels in Figure 2.5 show the histogram of the connection duration for charging sessions for two separate groups of users, in order to confirm the mentioned assumptions. Figure 2.5D shows the situation from Figure 2.4B showing merely sessions of users with more than 50 sessions. Figures 2.5A- 2.5C show the histogram with lower value of boundary for sessions' count of the user. For occasional users, the majority of sessions last less than 4-5 hours, as expected. On the other hand, we assumed this trend just for users with small count of sessions, while also sessions in Figure 2.5D behave similarly.

At this point, we cannot state the impact of occasional users on the accuracy as well as we

do not interpret and use this information, since it is not the objective of this work. Possible future research should also include clustering of the input data from mentioned perception.

To sum up, these are the main points possibly useful to propose the features:

- Aggregation of the features by users' identification is promising.
- Aggregation of the features by charging point should be beneficial.
- Statistical features proposed for sessions lasting more than time offset most likely improve the prediction accuracy.

2.2 Feature engineering

To characterise a charging session, we compile a set of features that can be organised in four groups: static features, features describing long-term charging history, features describing short-term charging history and online features (visible in Figure 2.6). Features included in the first three groups capture developments taking place before the start of the charging session upon which the prediction is made, and they are designed by considering previous studies [4, 8, 5]. The online features capture the progress from the start of the charging session until the time when the prediction is made.

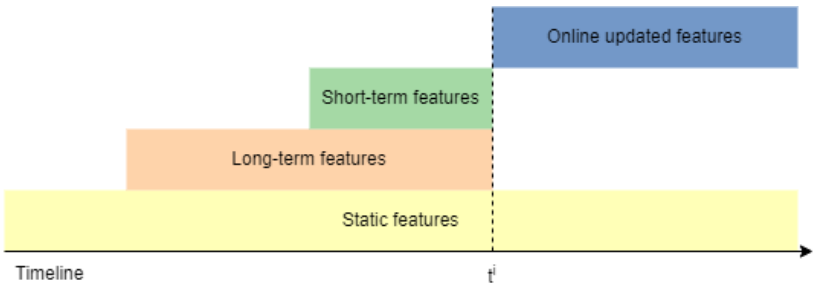

Figure 2.6: Graphical illustration of the features' categories.

2.2.1 Static features

These features take a constant value for all sessions associated with a station or a user:

- first two letters of the station label encoding its type (modelled as a categorical variable),
- longitude and latitude of the station,
- maximum charging power estimated as the minimum of the user maximum power and station maximum power.

2.2.2 Features describing long-term charging history

Features to capture characteristics of charging sessions in a long-term by aggregated statistics:

- mean, minimal and maximal values of the total charged energy, connection duration and charge duration (all values are calculated for both, charging sessions previously made by a user and charging sessions previously taking place at a charging station),
- relative frequency of sessions that lasted more than is the current connection duration (the value is calculated for both, for a user and for a charging station).

2.2.3 Features describing short-term charging history

Features calculated for n most recent days or n most recent sessions, to capture the short-term charging history:

- the mean value of the charged energy, the mean value of the connection duration and the count of sessions considering last day (week) for each station,
- the mean values of the energy consumption and connection duration in the last 1, 5, 10 sessions for each user.

2.2.4 Online updated features

Features that capture the progress of the charging session since it has started:

- current hour of the day, weekday and month (modelled as categorical variables),
- total charged energy since the start of the session,

- charged energy in the last 15, 30 and 60 minutes,
- connection duration since the beginning of the session.

2.3 Feature processing

In this section, we describe how we obtain the values of upper mentioned features. To avoid the knowledge from the future sessions, we use the rolling mechanism. The rolling features are calculated from previous events (in our case, sessions), instead of calculate features from whole dataset. In the other words, features for the k-th session in the dataset are figured out from previous $k-1$ sessions. Hence, features describing long-term charging history require historical sessions, denoted as warm-up set, to obtain the features.

Algorithm 1 Pseudocode for feature processing

Require: $\mathbf{S}_{warmup}$, $\mathbf{S}_{others}$

```
Create static features for S_others
Create short-term features for S_others
Create statistics for users from S_warmup
Create statistics for charging points from S_warmup
for all sessions in S_others do
    Create update offsets
    for all update offsets do
        Create observation
        Append observation to the list
    end for all
    Update users' statistics
    Update charging points' statistics
end for all
Convert list of observations into dataframe
Return dataframe
```

To meet the mentioned condition, we have to iterate all sessions and after every iteration

we have to update the statistics. Moreover, in our scenario, we divide each session into multiple observations. Each observation represents features for the i-th update of the prediction for k-th session. In practise, update of the statistics during charging process from another session is possible merely for predictors aggregated by charging point. Obviously, user itself cannot change the statistics for features used for prediction update in the next time offset. Therefore, it can happen solely for features grouped by charging point, namely for mean, minimal and maximal total charged energy, connection duration and charge duration. Hence, the correct process would divide sessions into observations representing time offsets, order observations by t_k^{arr} and iterate these observations. This brings increased computational burden. Since influence of the statistics for observation from another observation is unlikely (it occurs merely for the charging point containing more than a single connector), we omit this situation to speed up the whole process. The pseudocode of feature processing is shown in Algorithm 1.

Firstly, the static features are calculated for all sessions. These features have nothing to do with rolling mechanism. Thus, implementation is vectorised and very effective. The first two letters from charging point name are subset, referring to the type of the charging station. Since the maximum charging power is unknown, we estimate it as the minimum of the user maximum power and station maximum power from the whole dataset. In practise, maximum power is known at the time of EV arrival.

Short-term predictors are created in two steps:

1. Calculation of the last sessions' mean energy and connection duration. In our scenario, we calculate values for last session, last 5 session and last 10 sessions. To optimise the code, we iterate through the session just once and store current values of the observed values in a class named LastSessions. When a new session appears, the oldest session is popped out from the queue of last sessions in the instance of the class. To decrease computational burden, the sums of energy and connection duration for all sessions are stored and available. Class updates the sums on a change of stored sessions. Early on the cycle, empty dictionaries for users' last sessions and charging points' last sessions are available. As the cycle unfolds, new instances of the LastSessions are inserted into dictionaries, using user identification and charging point name as the keys. This solution with dictionaries of users' and charging points' last sessions we find more efficient

and faster than parallel iterations for each user's sessions and for each charging point's sessions, since the complexity is linear (we assume complexity = $O(1)$ for insertion into the dictionary).

2. For calculation of the energy, connection duration and session count from last day and week for each station, function rolling was used [45]. For sessions grouped by charging point, we apply this function with frequency one day and one week on the date column, closed from the left side. Since pandas functions are highly optimised, the performance is not computationally demanding.

For every user and station, we store historical statistics in the dictionary of lists of pandas dataframes. Every list represents statistics for user or charging station and contains as many dataframes as is the count of updates for user or charging point. Each dataframe contains minimal value, maximal value, the sum of the records and count of the records for connection duration, charge time and total charged energy, according to Section 2.2. We store sum of the records, instead of mean value due to computational simplicity. Relative frequency of the sessions lasting more the time offset for index i can be calculated as number of the records for time offset with index i divided by time of the records for time offset with index 0.

For each user or charging station, we get the statistics from the dictionary. If a record does not exist, a list with empty statistics is created and inserted into the dictionary. Index of update i is also an index into the list of dataframes. In dataframe, column name refers to required value. These statistics are updated after every iteration as we consider rolling features. While the connection duration of the current session is lower than the time offset of update i, we update all statistics for dataframe on the index i in the list.

The inner cycle in Algorithm 1 creates multiple observations for every session corresponding to given time offsets. For each observation, we update online updated features according to time offset. The time of the arrival of the EV t^{arr} is incremented by the time offset in order to obtain current day, weekday and month. Connection duration since the beginning of the session equals to the actual time offset. Charging rate is assessed to the total charged energy divided by charge duration, since we model uniform charging rate throughout the session. Thus, the charged energy since the beginning of the session is calculated as charging rate

multiplied by actual time offset. Note, that time offset can be higher than charge duration, engendering higher value of charged energy since the beginning of the session than total charged energy. Therefore, we have to cap the value of charged energy to the total charged energy. Finally, charge energy in last 15, 30 and 60 minutes is calculated by the separate function. This function is generic and requires interval in hours (in our case, 0.25, 0.5, 1 hour), actual time offset, total charge duration and assumed charging rate. There are multiple cases to consider:

- Current time offset is lower or equal to total charge duration and interval is lower or equal to actual time offset (Figure 2.7A). The function returns interval multiplied by charging rate.
- Current time offset is lower or equal to total charge duration and interval is higher than actual time offset (Figure 2.7B). The function returns time offset multiplied by charging rate, i.e. total charged energy since the beginning of the session.
- Current time offset is higher than total charge duration. Variable start takes the value of time offset retracted by interval. If start is lower than total charge duration (Figure 2.7C), we recalculate an interval to total charge duration retracted by start variable and function returns this interval multiplied by charging rate; otherwise charge energy in interval is zero (Figure 2.7D).

The described procedure provides features' creation with linear complexity, thus without significant computational burden. Despite that, duration of the whole process increase with raising number of time offsets, i.e. number of updates. The biggest disadvantage is that the dataset is created priorly and thus with change of any time offset, the recalculation of the whole dataset is required. To handle this, dataset with very low value of δ can be created and then just observations corresponding required time offsets are subset.

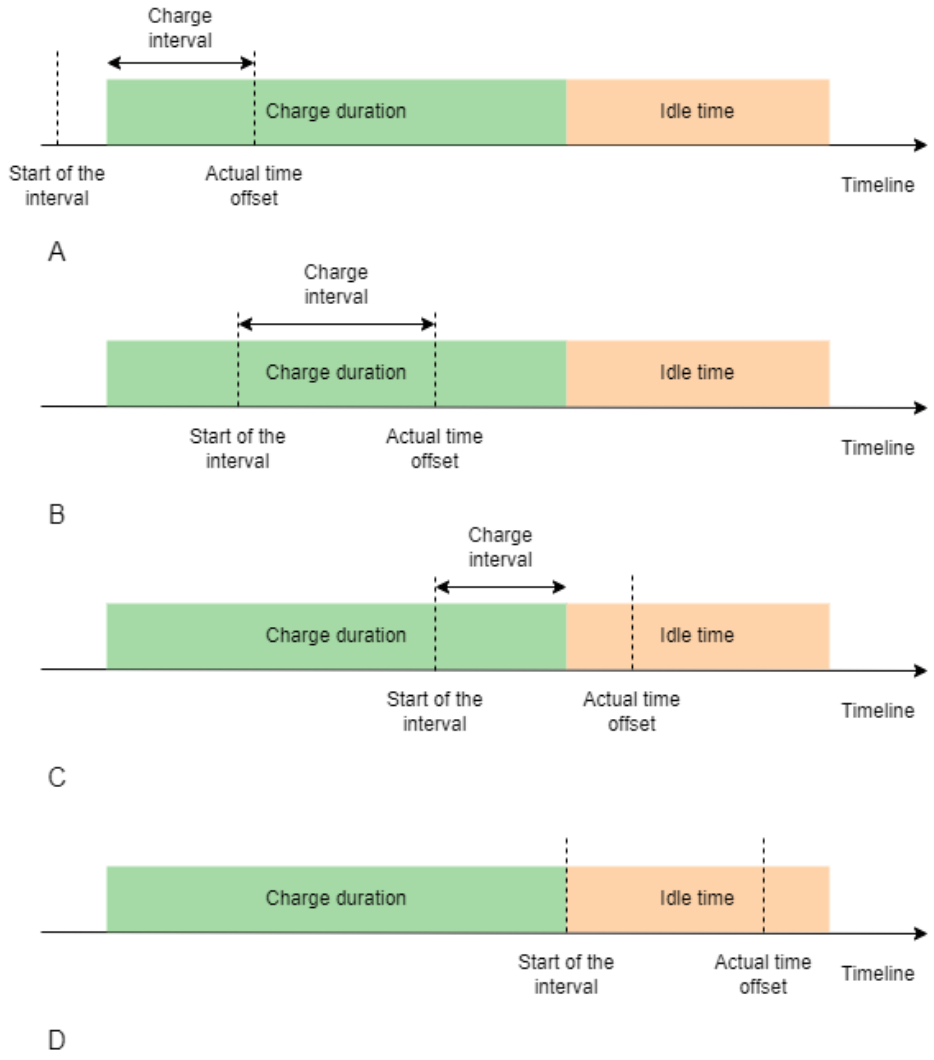

Figure 2.7: Graphical illustration of the cases for calculation of charged energy in interval. Panel A shows situation, when the interval is higher than actual time offsets. Panel B shows the case of the whole interval within the session charge duration. Both, idle time and charge time, are included in the interval in panel C. Panel D shows, when no energy is charged in the interval.

Chapter 3

Methods and model construction

The following chapter is divided into three main logical parts. The first part introduces used methods and update strategies. The second part handles the construction of proposed models and describes implementation details. Finally, we describe how we evaluate the accuracy of the predictions in order to applications, such as smart charging and charging point availability.

3.1 Methods

In this section, we introduce methods used for predictions of EV connection duration. Firstly, naive methods are described, providing fair benchmark for LightGBM (LGBM) method. We also introduce Bayesian optimisation used for hyperparameters tuning.

3.1.1 Naive models

Two naive models are applied to assess the performance of the proposed prediction models. First is the mean connection duration of a given user as a static prediction approach (mean-static). To provide a fair benchmark for the updated models, we calculate the mean connection duration of all the user's sessions lasting longer than the considered session (mean-updated).

For both, mean static and mean updated we can use prepared statistics from predictors processing, since for every session they are figured out solely from prior sessions stored in the features' values of the observation. Therefore, mean-static and mean-updated predict

connection duration using upper mentioned mean connection duration for a user of sessions lasting more than connection duration since the beginning of the current session. The only difference is that mean-static is considered just in time of arrival of the EV and mean-updated is considered in every defined time offset.

3.1.2 LightGBM

We use LGBM [46] as the prediction method, which is a state-of-the-art Gradient Boosted Regression Trees (GBRT) [47, p. 359–361] implementation. LGBM coping with time consumption during estimating gain throughout all possible split points by two novel techniques: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB), engendering that LGBM speeds up the training process of conventional GBRT by up to over 20 times while achieving almost the same accuracy. In our scenario this is a large advantage as we re-train the model several times.

Python Scikit-learn application interface for LGBM [48] provides python implementation of the algorithm, with conventional functions, such as fit and predict. Implementation offers good accuracy with integer-encoded categorical features [49], since applies Fisher [50] in order to find the optimal split over categories. Thus, LGBM often performs better with categorical features than one-hot encoding. Our features meet integer-encoded categorical features, except the first two letters of the charging point name. Hence, we transform the first two letters of the charging point name to a contiguous range of integers started from zero, as recommended.

3.1.3 Bayesian optimisation

Bayesian optimisation is an iterative algorithm, based on a probabilistic surrogate model and an acquisition function. To decide which point to evaluate next, the expected improvement

$$\mathbf{E}[\mathbf{I}(\lambda)] = \mathbf{E}[max(f_{min} - y, 0)] \tag{3.1}$$

is mainly used, since it can be computed in closed form if the model prediction y at configuration λ follows a normal distribution [51, p. 9]. For paper needs, we use Tree-structured

Parzen Estimator Approach (TPE) modelling conditional probability by transforming configuration space, described by a graph-structured generative process, replacing the distributions of the configuration prior with non-parametric densities to facilitate the optimization of expected improvement [52].

GetBestParamsBayesian function implements general case of Bayesian optimisation using hyperopt package, namely fmin function, space_eval and hp (containing TPE approach). Fmin function requires:

- Objective function returning score for configuration of the hyperparameters (black box function).
- A dictionary with name of the parameter and possible discrete values or interval called space.
- Algorithm for searching within the space of hyperparameters. As we mentioned upper, we use TPE approach.
- Upper bound for count of iterations, called max_evals. The count of iterations means how many times is the model built and evaluated, i.e. how many times the objective function is called.

In our scenario, the objective function uses the current configuration of the hyperparameters and creates the model. To figure out the score of the model, cross validation from sklearn.model_selection package is used. Due to rolling features, we have to use evaluation on rolling features origin instead of conventional cross validation because the origin on which the features are based roll forward in time [53]. To split data, we use a variation of k-fold validation TimeSeriesSplit from Scikit-learn package [54], which returns the first k folds as a training set and the $k+1$-th fold as a test set. Figure 3.1 shows how evaluation on rolling predictors origin works.

Finally, the objective function returns a negative score value, since fmin function tries to minimise the objective value.

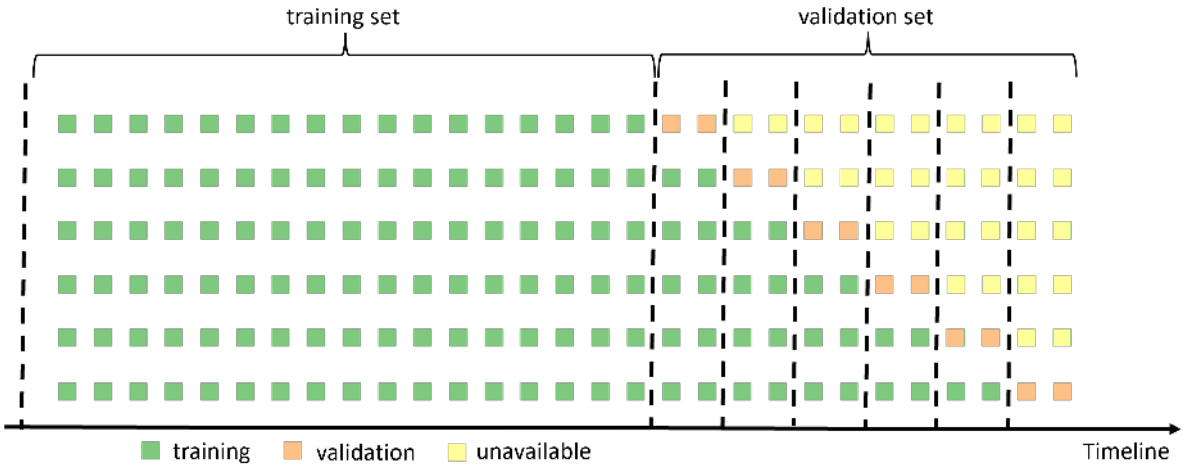

Figure 3.1: Evaluation on rolling features origin using k-fold cross validation with time series split ($k = 6$).

3.2 Update strategies

To increase the overall model accuracy, we update the predictions discretely in time, as in [16]. We propose two basic approaches to handle the prediction updates. The first we, denote as synchronous approach and it updates all the sessions in the same time. The second handles each session's updates individually and we denote it as asynchronous approach.

Synchronous approach regularly updates the predictions based on a global clock, independently of EV arrivals Figure 3.2 (A). On the contrary, asynchronous approach updates every session individually based on the arrival time of the EV (figure 3.2, panel B). From the perspective of training the prediction models, asynchronous approach brings an advantage of updating the predictions of each session within the same periods from the session's start time. This allows us to reshape the data into a more convenient format. We aggregate the features by the times of updates and hence generate more advanced features. Moreover, we group the observations by the prediction update time. All these transformations bring additional benefit in speeding up the computations in model training.

For instance, let use two sessions. Session A starts at 8:15 and lasts 4.5 hours, i.e. finishes at 12:45. Session B starts at 8:45 and lasts 2 hours, i.e. finishes at 10:45. Let suppose regular

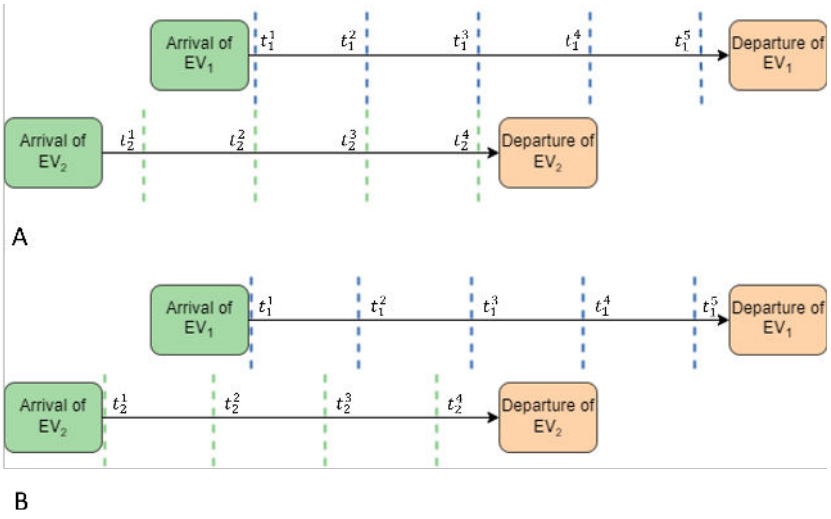

Figure 3.2: Panel A: Synchronous updated predictions. Panel B: Asynchronous updated predictions. Each panel contains two separate sessions for EV_1 and EV_2. Arrival of EV means start of the session and departure of the EV means end of the session. Each session k contains several times of update t_k^i.

update with $\delta = 1$ hour. System based on synchronous approach ticks regularly at a time 0:00, 1:00, ..., 23:00, 0:00, 1:00 and so on. Therefore, time offsets for session A are in time offsets 0.75, 1.75, 2.75 and 3.75 corresponding to the global ticks 9:00, 10:00, 11:00 and 12:00. Similarly, session B updates prediction in time offsets 0.25 and 1.25 corresponding to global ticks at 9:00 and 10:00 hours. On the contrary, system based on asynchronous approach works differently. Both sessions start with prediction on arrival and then use regular $\delta = 1$ to calculate time offsets of the prediction update. Thus, session A has got time offsets equal to 0, 1, 2, 3 and 4 corresponding to times 8:15, 9:15, 10:15, 11:15 and 12:15. Session B updates prediction with time offsets 0, 1, 2 corresponding to times 8:45, 9:45 and 10:45.

System based on synchronous updates can be centralised into a single point of the charging infrastructure. An obvious disadvantage is that all information simultaneously flow to the central point, instantly rising load to the communication network. System with asynchronous updates loads the network more uniformly, since each session is updated based on the EV arrival time, i.e. in different times.

From the upper mentioned reasons, in this paper we use asynchronous discrete update.

Asynchronous discrete update allows us to propose more approaches to build the model.

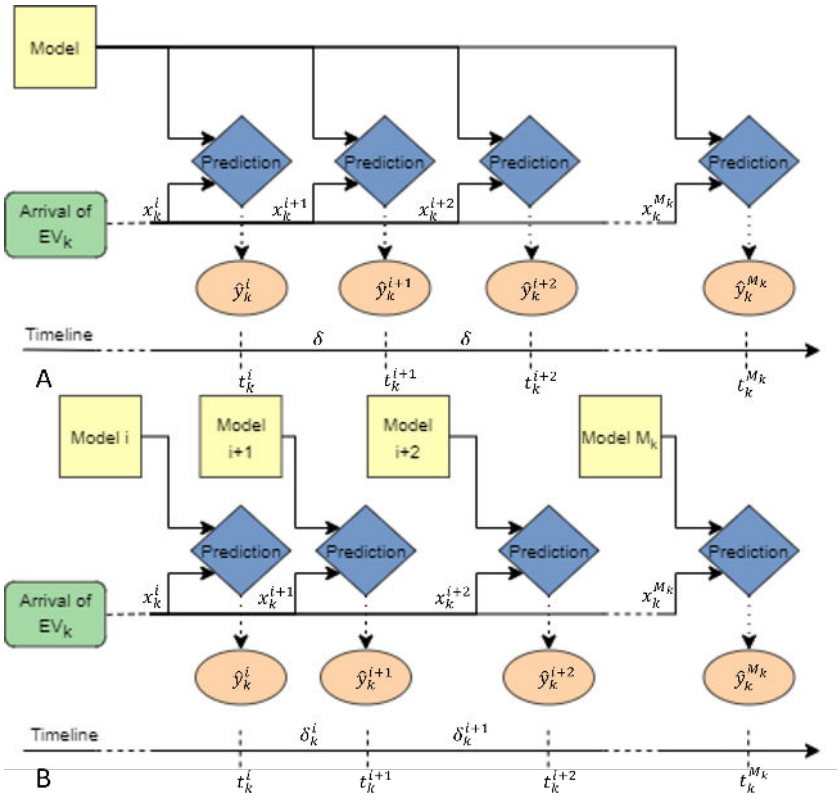

Figure 3.3: Panel A: A schematic illustration of the single model approach with regular update strategy. Panel B: A schematic illustration of the multimodel approach with irregular strategy. For every time of update t_k^i for $i = 1,..,M_k$, the prediction $\hat{y}_k^i$ is made using vector of feature values x_k^i. Duration until next planned update or EV departure in i-th time of update is denoted as δ_k^i. In regular update strategy, $\delta_k^i = \delta$ for every $i \neq M_k$.

- Static approach - the connection duration is predicted only once, when the EV is plugged in.
- A single-model - all predictions during the sessions are made by the single model (Figure 3.3 panel A).
- A multi-model - prediction model is divided into multiple 'sub-models', where every model belongs to the appropriate update time. Each of the models can be trained via different prediction methods (Figure 3.3 panel B). For every time of update, different

sessions are active, i.e., have longer connection duration than the time offset of the update.

The static approach uses just observations at the time of arrival t^{arr} of the EV. These observations contain values for all features as described in Section 2.2, however online features and features aggregated by current duration are not meaningful. The single-model uses the whole dataset to train a single model and predict connection duration for each session in every time offset of prediction update. Current duration helps the model to predict the target variable, since this feature contains information about minimal duration (sessions cannot finish before observations' current duration). Finally, the multi-model is similar to static-model in the usage of the sessions with same time offset, i.e. current duration. Feature current duration is needless, since it is equal for all sessions in the dataset.

Let use session A with time offsets of update 0,1,2 and 4, session B with time offsets of update 0,1 and 2 and session 3 with time offsets of update 0,1, ..., 16. For static approach, just 3 observations with time offsets equal to 0 are used. For single-model and multi-model all 24 observations are used. For single-model observations are used at once, while multi-model creates sub-models for every time offset, thus 17 sub-models with appropriate observations. At the time of arrival t^{arr} of the EV (time offset equals to 0), multi-model and static approach use the same set of observations.

In general, synchronous approach predicts at the arrival of the EV. Therefore, we consider this approach as extension to static approach, since the first prediction (at the arrival of the EV) would be the same. Later on, predictions are updated in the planned time offsets, that would improve the former prediction. Hence, we assume that overall accuracy has to be better or equal to static approach, except the case of worsening of the former predictions in time offsets. Moreover, for asynchronous approach, we can use regular or irregular updates.

3.2.1 Regular updates

For regular updates is the update step set to the same value during the whole session (figure 3.3 panel A). For the whole system, the general value of δ valid for each session is determined.

3.2.2 Irregular updates

Within the irregular updates, the length of the update step can decrease or increase through the session (figure 3.3 panel B). The irregular update adapts the times of update with the goal to minimise the error of the prediction. For instance, we can utilise shorter steps in the periods of higher importance and vice versa. However, the risk of overfitting increases. Moreover, for irregular updates, we have to adjust the traditional prediction accuracy measures.

Let use the upper mentioned example with session A starting at 8:15 and lasting 4.5 hours and session B starting at 8:45 and lasting 2 hours. Previously, the example describes synchronous and asynchronous system with regular step of update $\delta = 1$ hour. Thus, for asynchronous system, time offsets are 0, 1, 2, 3 and 4 corresponding to times 8:15, 9:15, 10:15, 11:15 and 12:15 for session A. Session B updates prediction with time offsets 0, 1, 2 representing times 8:45, 9:45 and 10:45. Concerning irregular update strategy, let use time offsets 0, 0.5, 1, 2, 3, 3.5 and 4 for whole system. For both, session A and session B, time offsets will be the values from determined time offsets until the session finishes. Hence, session A updates prediction at 8:15, 8:45, 9:15, 10:15, 11:15 11:45 and 12:15. Similarly, session B updates prediction at 8:45, 9:15, 9:45 and 10:45.

In general, determination of the same time offsets for the whole system is not required. Therefore, we can use for every user, charging point or session own regular/irregular time offsets. This allows us to utilise the time offsets for particular day hour, group of users or charging points or group of session grouped by general patterns, as it was in [6, 7]. At this point, we omit these possibilities to emphasise the main goals of this paper, even though it would be beneficial for prediction accuracy.

3.3 Construction of models

To avoid peeking, i.e. using test-set performance to both choose a hypothesis and evaluate it [55, p. 709], we use the test set as hold out. For evaluating the different configurations of the proposed models, we use the validation set [56]. Thus, the whole dataset was divided into training set, validation set and test set. Additionally, the warm-up set was created, since warm up is required for the feature engineering process to create default statistics for users and

charging points. To find the appropriate size of warm-up set, we have to consider a trade-off. On the one hand, we need sufficient size to include as many charging points as possible. On the other hand, keeping as many sessions as possible for training set and test set is necessary. Since we have a large dataset (up to a million charging sessions), we can use the sessions in 2016 for warm-up. Session from 2017 are used for training set and the earliest 50k sessions from 2018 are used for validation set. The rest of the 2018 sessions are used for the test set.

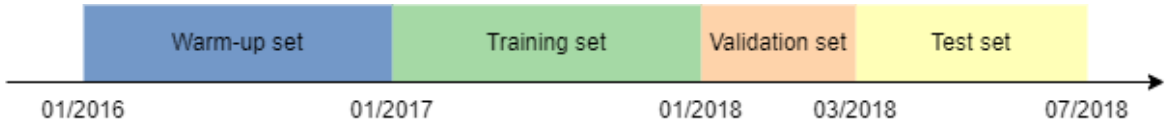

Figure 3.4: Illustration of the warm-up set, training set, validation set and test set on timeline.

The class Predictors stores features and particular datasets as well as provides some basic functions required to simplify experiments. Data divided into training, validation and test set enter into constructor as parameters. For regular update, the features' dataframe contains observations with very low δ in order to speed up the creation of features' dataframe as we mentioned in Section 2.3. Therefore, for large parameter δ, we have to subset just proper observations, priorly in the constructor. Consequently, datasets are joint into one dataframe to process the categorical features, since we need to ensure that features created by one-hot encoding will be in every set. Later on, sessions are divided back to original data sets. The same practise is used for marking the features' columns as categorical, necessary for LGBM algorithm, since it performs well on categorical features without one-hot encoding. Moreover, static-wise sets are created from observations with current duration equal to 0, representing dataframes for static approach. Note, that the proportion of size of sets could change, while origin proportion depends on sessions' connection duration in dataframes.

3.3.1 Single-model

For training a Single-model, LGBM was used. In practise, we implement single-model class to simplify code for experiments and figures creation. An instance of single-model contains

valid columns, i.e. active features for training and prediction, and current method. In our case, columns correspond to features mentioned in Section 2.2.

3.3.2 Multi-model

Due to simplicity, LGBM was used for every sub-model in Multi-model approach and both, for regular and irregular strategy.

Construction of the regular Multi-model depends on the δ which is directly proportional to the count of models. Each of the models is trained separately on its own dataset, containing just observations from active sessions with target variable y_k and updated features x_k.

Implementation of the Multi-model class has to afford basic functions, likewise Scikit-learn methods, as well as some extensions essential for multi-model approach:

- Functions - fit and predict.
- Feature importances dataframe.
- Bayesian hyperparameters tuning.

Alike Single-model, Multi-model instance requires valid columns and method for prediction. In our implementation, multiple types of method for a multi-model are not allowed. Contrary to Single-model, Multi-model class contains multiple models in an array, each for a particular time offset. Additionally, Multi-model allows optional parameters for method, since it clones method multiple times and consequently compiles methods with appropriate parameters. Concerning regular update strategy, the parameter δ determines the size of the array with models.

Fit and predict functions work similarly to state-of-art algorithms provided by conventional packages. Dataframe with all observations and all features is an input for these functions. Both, fit and predict, iterate observations grouped by current duration column. Fit function calculates index in array of models and fit model using group of observation. Predict function iterates groups with goal to predict connection duration for each observation within the group and create dataframe with session's identification number, current duration,

predicted connection duration, real connection duration (target variable) and residual, i.e. the difference between predicted connection duration and real connection duration.

Since we use LGBM as a prediction method and this method provides feature importances already used for feature selection, for instance for environmental data [57], where LGBM responds well to feature selection. We can use this amenity to create a table with feature importances for Multi-model. This table consists of all valid columns representing features' names and all time offsets for update prediction of the connection duration. The main goal is to compare importance in different time offsets, if any appears.

A function for hyperparameter optimization requires the following parameters:

- Training set - dataframe used for training process.
- Validation set - set for k-fold cross validation.
- Method - prediction method, in our scenario we use just LGBM method.
- Space - dictionary with name of the parameter and possible discrete values or interval.
- Number of splits - number of folds for cross validation.
- Scoring method - scoring method evaluating the performance and accuracy of the prediction.
- maximal count of evaluations - number of iterations for Bayesian optimisation.

Hyperparameters for each model are optimised separately. Therefore, we group observations in training and validation sets by current duration. For each group, we calculate test size corresponding to validation subset and append both sets into single dataframe and use it for k-fold Bayesian hyperparameters' optimisation. Finally, we train *i*-th model using the best parameters from Bayesian hyperparameters' optimisation.

Concerning irregular update strategy, the most suitable values of update times for the Multi-model are found by adjusted hyperparameter optimisation. There are many ways how to model time offsets as hyperparameters. The task is to create space for hyperparameters optimisation from $[0, 24]$ interval, with the goal to construct time offsets from this space. Divide

the $[0, 24]$ interval into more intervals with own δ is the first option. This represents easy way how to propose hyperparameters tuning and transform configuration of the hyperparameters to time offsets. On the other hand, proposed option mitigate the flexibility of the time offsets, since each interval has the same value of δ for every time offset. Moreover, time offsets in intervals' boundaries are compulsory. Hence, time offsets are considered as hyperparameters from the uniform discrete distribution $\mathscr{U}_{\{0,24\}}$, engendering the maximal flexibility for adjusting the time offsets. Note, that more time offsets can obtain the same value; however, it is unlikely. The biggest advantage is that we can control the count of updates of the prediction, while in the case of dividing the whole interval on the multiple subintervals it is not possible.

Many computations with various time offsets for update are considered, and creation of features for every configuration of time offsets is time-consuming (we would follow to process described in Section 2.3). Therefore, we omit the statistics describing long-term charging history and create online features not in advance, but during training process, literally online. Since to calculate error value for different update times is a computationally expensive black-box function, Bayesian hyperparameters' optimisation finds best suited time offsets for update of the predictions, using reduced dataframe. For final selection of time offsets for update of the predictions, a comprehensive set of features is created priorly by a conventional process described in Section 2.3.

VariableStep class implements Multi-model irregular approach. While Multi-model class uses for both, fit and predict functions, comprehensive training and test set, VariableStep class extends these functions and updates features after every iteration, i.e. after fitting or predicting in particular time offset. Firstly, from current observations are selected and copied just sessions with connection duration higher or equal to current time offset corresponding to observation of active sessions. The current duration column is updated to the current time offset. Finally, we have to handle charged energy update. There are many ways how to assess charged energy, but we have to consider computational demands. Thus, we use vectorisation provided by pandas dataframe to figure out the charge energy as the column charging rate multiplied by current duration. The average charging rate for each session is obtained as the total energy divided by charge time. Since the current duration can be higher than charge duration, min function figures out the minimum from charged energy and total charged energy.

Similarly, we adjust the objective function in algorithm for Bayesian hyperparameters' optimisation from Subsection 3.1.3 and call this function as getBestTimeOffsets. The function creates time offsets using parameters from the argument to build the VariableStep multi-model instance. Multi-model fits models on the training set and evaluate the prediction accuracy on the validation set. In our scenario, the function returns weighted mean (see Section 3.5).

Accordingly, we can illustrate the process as follows. For n updates we need $n-1$ time offsets because the first prediction is made in time offset equal to 0. Hence, we create a dictionary with $n-1$ parameters as a hyperparameters' space. The key of the record is index of the update and interval of values is created using hp.quniform function from hyperopt package. Input for the function contains:

- Lower bound of the interval. We recommend using the value of granularity as a lower bound to avoid the multiple update in time of arrival.
- Upper bound of the interval. In our scenario, the upper bound is 24 representing maximal duration of the session.
- Granularity of the created space. For this paper needs, we use granularity 0.01 and so 6 minutes as a minimal δ.

In every iteration of Bayesian optimisation, different $n-1$ values of time offset entry into objective function, and they are extended by 0 value. Based on ascending ordered time offsets, an instance of VariableStep is created and evaluated. Best parameters from the Bayesian optimisation representing the most suitable time offsets.

3.4 LightGBM hyperparameters

LGBM uses the leaf-wise tree growth algorithm, while many other popular tools use depth-wise tree growth. This brings faster convergence, besides possibility of the over-fitting if not used with the appropriate parameters [58].

According to the LGBM documentation, we tune some important parameters. Parameter num_leaves controls the complexity of the tree model, as it determines the maximal number of

leaves in a single tree. Hence, it can both increase the accuracy and induce over-fitting. Small values of the number of leaves speeds up the training process. Parameter min_data_in_leaf significantly prevents over-fitting, while the parameter's value depends on the number of training samples and number of leaves. For large dataset, setting it to hundreds or thousands is sufficient. Since LGBM uses decision trees as the learners, the num_iterations parameter controls the number of trees for boosting. With changing the num_iterations, change of the learning_rate is recommended. This will not have any impact on training time, but it would impact the accuracy. In general, reducing of the num_iterations, should be followed by increase learning_rate. Finally, large max_bin causes better accuracy, but can increase the computational burden.

3.5 Error measures

Since each session can be updated multiple times depending on the update granularity and session duration, we propose the following prediction error measures.

The weighted mean absolute error [59] is adapted to our scenario as the average of the mean weighted absolute error throughout all updates from all sessions

$$wMAE = \frac{1}{N}\sum_{k=1}^{N}\frac{\sum_{i=1}^{M_k}|y_k - \hat{y}_k^i| * \delta_k^i}{\sum_{i=1}^{M_k}\delta_k^i}, \tag{3.2}$$

where y_k is the target variable, $\hat{y}_k^i$ is the predicted duration for session k in the update time t_k^i, N is the count of all sessions, M_k is the count of updates for session k and δ_k^i is the duration until next planned update or EV departure in i-th update for session k. For every session, $\sum_{i=1}^{M_k}\delta_k^i$ equals to target variable y_k. For regular update, δ_k^i equals to δ for every session k and $i \neq M_k$ and for every session k, $\delta_k^{M_k}$ is the minimum of regular updating step and time to the vehicle departure.

Since, wMAE function is used multiple times (especially for hyperparameters' optimisation), low computational burden is necessary. To vectorise the function, we modify the calculation from (3.2) to

$$\frac{\sum_{i=1}^{M_k} |y_k - \hat{y}_k^i| * \delta_k^i}{\sum_{i=1}^{M_k} \delta_k^i} = \frac{1}{y_k} * \sum_{i=1}^{M_k} |y_k - \hat{y}_k^i| * \delta_k^i = \sum_{i=1}^{M_k} \frac{|y_k - \hat{y}_k^i| * \delta_k^i}{y_k} \tag{3.3}$$

for each session, since $\sum_{i=1}^{M_k} \delta_k^i$ equals to connection duration y_k. Hence, for each observation, we have to calculate

$$\frac{|y_k - \hat{y}_k^i| * \delta_k^i}{y_k} = \frac{|r| * d}{y_k}, \tag{3.4}$$

where r is the residual and y_k is the real connection duration, both returned in dataframe by function predict, as we mentioned in Subsection 3.3.2. Error duration d can be calculated as the minimum value from δ_k^i and remaining duration $(y_k - t^i)$.

Moreover, according to (3.2), we can consider the benefit of the updating approach as in the Figure 3.5A-Figure 3.5C. The prediction made by static approach holds during the whole session. Therefore, we can consider the accuracy enhancement of the update approach as the difference area in between static and update approach.

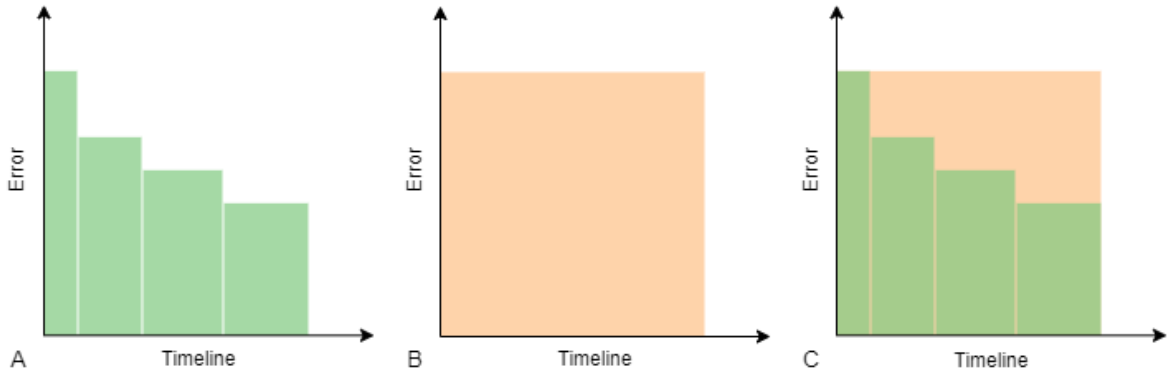

Figure 3.5: Geometrical intuition of wMAE measure for session's predictions using update (panel A) and static approach (panel B). The comparison of the static and update approach is in the panel C

To consider and compare performance in particular time of update t^i, an additional metric

$$MAE_{t^i} = \frac{1}{N} \sum_{k=1}^{N} |y_k - \hat{y}_k^i|. \tag{3.5}$$

is proposed. Only the active sessions are considered. To evaluate predictions at the time of EV arrival, we apply MAE_0, i.e. at MAE time $t^1 = 0$. To implement the (3.5), we group observations by time offset. To calculate conventional MAE from these observations is sufficient.

3.6 Smart charging scheme

According to [60, p. 106] we can optimise the charging session in order to three different objectives: the time (by postponing the charging), the speed (by increasing or decreasing the charging power), and the charging direction (by switching between charging and discharging of an EV throughout the session). In this section, we consider a baseline scheme that represents business-as-usual (BaU) case of uncontrolled charging and ToU smart charging scheme optimising the charging time. The schemes are introduced together with main criteria enabling the quantification of the consequences of under- and over-estimation of the charging duration:

- E^n represents demanded but not delivered energy,
- E^p is the energy charged in peak period for peak price,
- E^o is the less expensive energy charged in offpeak periods.

Total charged energy for each session k obtained from the EVnetNL dataset equals to the sum of upper defined criteria, thus $E = E^n + E^p + E^o$.

3.6.1 Business-as-usual charging

BaU charging scheme represents the baseline, where no smart charging scheme is employed. The charging is initiated at the time t^{arr} when the driver plugs in the vehicle to the station and continues until the time t^{char}. At this point, the vehicle is fully charged or the driver unplugs it. In practise, the charging is performed with nearly uniform power. Therefore, we estimate the charging power uniformly and for computational experiments, it equals to total energy divided by charge duration. The BaU is shown in Figure 3.6B.

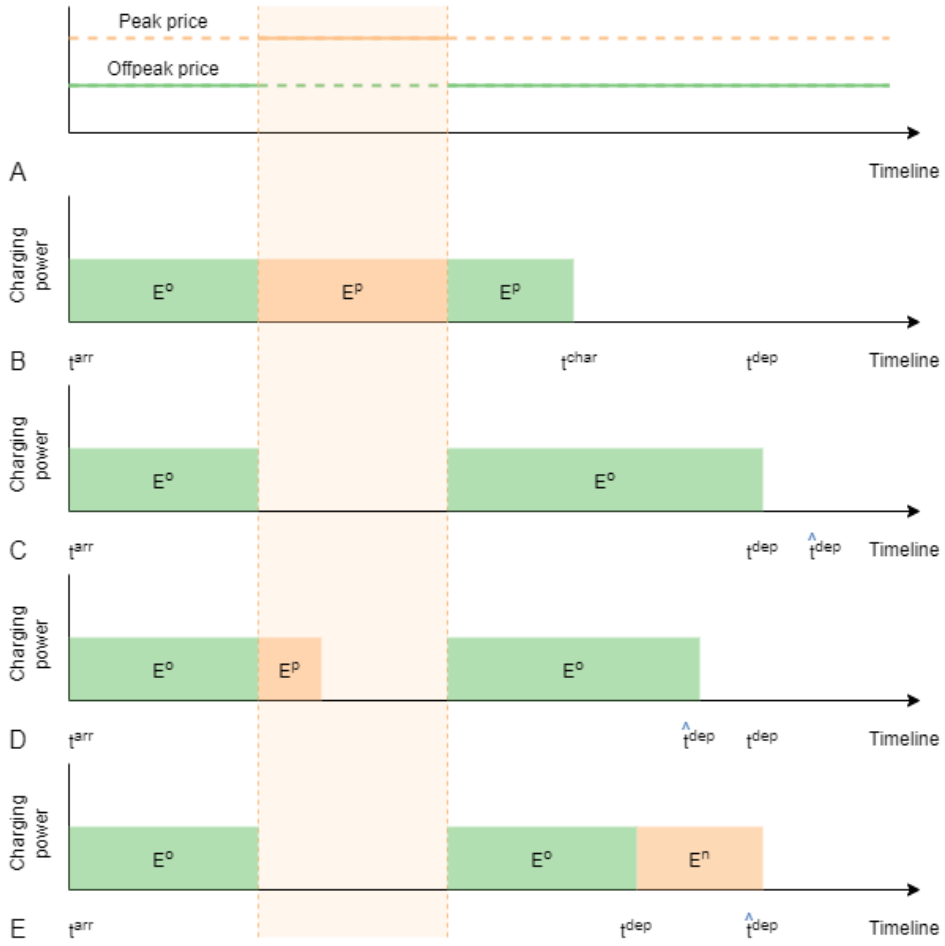

Figure 3.6: Comparison of the BaU, the UP and the ToU smart charging schemes. Panel A: Price signal. Panel B: BaU charging scheme. Panel C: ToU charging scheme, where the high price period is fully avoided. Panel D: ToU charging scheme, where the high price period is only partially avoided. Panel E: ToU charging scheme, where the high price period is entirely avoided, but less energy than by the BaU scheme is provided to the vehicle due to the overestimation of the connection duration.

Note, that from definition, BaU avoid E^n, since all demanded energy is charged in charge time, i.e. $E = E^p + E^o$.

3.6.2 Time-of-use smart charging

The ToU smart charging scheme is based on the price signal that varies over the predetermined price periods of the day in response to the power grid state [24, 61]. In general, the price grows together with the load. To simplify, in this paper, we consider only peak price level and off-peak price level as is visible in Figure 3.6A. Moreover, we assume, that EV can be charged at maximum possible power represented by the feature denoted as maximal possible power (see Section 2.2). The ToU scheme requires the connection duration priorly, thus an estimated value of connection duration is necessary. In order to minimise charging in peak period, the charging takes place in the peak price period only if the EV cannot be fully charged in the off-peak price period.

For ToU both, over-estimation and under-estimation of the connection duration, influence differently the criteria E^n, E^p and E^o. Estimation of the $\hat{t}^{dep}$ approximately equalled to true departure t^{dep} may allow avoiding peak price periods completely, as is visible in Figure 3.6C. On the contrary, panel D in Figure 3.6 displays the underestimation of the connection duration, hence $\hat{t}^{dep} < t^{dep}$. This leads to charging in the peak price period, even though it is not necessary. Despite that, charging is still more beneficial than BaU charging. Finally, when the vehicle is unplugged significantly earlier than estimated, the charging may be planned as well after the true departure t^{dep} of the EV. This brings demanded but not charged energy E^n (see Figure 3.6E).

3.6.3 Implementation

The whole implementation of smart charging schemes is vectorised, engendering very fast computations within tenths of seconds also for large data. The critical point to be vectorised is the calculation of the overlay for charging session and peak periods. For each session, we can easily figure out the start and end, represented by hour of the day. Since we cupped the connection duration by 24 hours, the boundary for maximal time of the end of the session is

48 hours (session start at the very end of the day and last 24 hours). Later on, we calculate the lower bound and upper bound as the maximum from start of the session and start of the peak period and minimum from end of the session and end of the peak period. Both operations are vectorised for data in pandas dataframe. Finally, the difference between upper bound and lower bound is calculated and clip by zero. Note, that we duplicate the peak period as well for the next day, i.e. period from 24 hours to 48 hours.

For BaU, we calculate the sum from the overlays for the session's charge period and for each peak period. The sum multiplied by charging rate represents energy charged in the peak period E^p. Consequently, $E^o = E - E^p$ and $E^n = 0$. We consider two cases for ToU smart charging scheme: for static approach and for updating approach.

Static ToU

Charging is planned at the time of arrival t^{arr}. The overlays are calculated from the t^{arr} and $\hat{t}^{dep}$ for each session and for each period. The maximal possible charge energy for both, peak and offpeak period are calculated. E^o is assessed as the minimum from total energy and maximal possible energy for peak period. Required energy in peak period is equal to the delta in-between the total energy E and E^o. Not charged energy E^n is calculated as $E - E^o - E^p$.

Updating ToU

The charging is planned in every time offset of prediction update, considering already charged energy. Firstly, two types of overlays are calculated:

- The overlays from the t^{arr} and $\hat{t}^{dep}$ for each session and for each peak period.
- The overlays from the t^{arr} and next update or $\hat{t}^{dep}$ for each session and for each peak period.

Later on, the observations are grouped by current duration. We iterate over groups ordered by current duration and for each group we calculate the criteria E^o and E^p similarly to static ToU. The calculations consider energy to charge instead of total energy. Energy to charge updated from previous group of observations. $E^n = 0$ for session having next update, otherwise E^n equals to energy to charge reduce by E^o and E^p of current observation.

3.7 Evaluation of the charging point availability

In Subsection 1.2.2 we claimed the contribution of this paper for charging point availability. According to [35], we consider the problem as the classification problem, predicting whether the sessions finish in the next 15 minutes. Additionally, we also consider 30 minutes period. The four possible scenarios can occur:

- The finish of the session is predicted and the session truly finishes represents a true positive value (TP).
- The finish of the session is predicted and the session does not finish represents a false negative value (FN).
- The finish of the session is not predicted and the session does not finish represents a true negative value (TN).
- The finish of the session is not predicted and the session finishes represents a false positive value (FP).

To evaluate the prediction, we can construct the confusion matrix and evaluate the sensitivity (known also as TP rate)

$$Sensitivity = \frac{TP}{P} \tag{3.6}$$

and specificity (known also as FN rate)

$$Specificity = \frac{TN}{N}. \tag{3.7}$$

Since we assume imbalanced dataset (more negative values occur), we use F1 score taking both FP and FN into account [62]. F1 can be calculated as

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{3.8}$$

where precision $= \frac{TP}{TP+FP}$ and recall $= \frac{TP}{TP+FN}$ [62].

Authors in [35] simulate the times, when the charging point availability is demanded and evaluate the predictions. To simulate the real demand, we use arrival times from the test set. We create a dataset with remaining duration and estimated remaining duration using predicted duration for the session in particular time offset. Moreover, we figure out the period within the prediction is actual. Later on, we iterate over the arrival times and for each we consider just active predictions. We use the remaining duration to assess whether the session finishes within the interval. Analogously, remaining estimated duration is used to estimate whether the session finishes within the interval. From these inputs we can calculate the state (TP, TN, FP and FN) of the session in arrival time. Finally, we calculate the sum of all states from active predictions and update the confusion matrix.

Chapter 4

Results

In this chapter, we compare prediction methods and evaluate the impact of the proposed update strategies. Furthermore, we examine their suitability for practical applications, such as smart charging and charging point availability.

In Section 4.1, we consider just the regular update strategy. Here we compare the performance of naive models with advanced models regularly updating predictions, and explore the influence of update frequency and observe the accuracy of models as a function of the time offset t^i. We examine the benefit of irregular update strategy as well as present the obtained time offsets t^i for the irregular Multi-model in Section 4.2.

While we used default values of hyperparameters for LGBM in Sections 4.1 and 4.2, we compare the previous results with results obtained using tuned models in Section 4.3. We do not tune all models, since it is computationally expensive and at the same time, it increases their accuracy just imperceptibly. Later, we use tuned models to apply the smart charging scheme as well as charging point availability scheme and evaluate the impact of proposed models and strategies. Finally, to search for the most relevant features and compare the importance of the features among the models, we show the most used features for splitting for LGBM.

4.1 Regular updates

In Table 4.1, we compare the performance of naive models with advanced models, updating predictions regularly every hour ($\delta = 1$). The results confirm the superiority of advanced models. The advantage of updating the predictions is already evident from comparing the Mean-static and Mean-updated models. On average, the error drops by almost 0.5 hour due to updated predictions. Similarly, when contrasting the Static-model with Single-model and Multi-model, the improvement is also about 0.5 hour. Despite the superiority of the Single-model to Static-model, the simple model is less accurate at the time of EV arrival expressed by MAE_0. This happens because the Single-model is also adapted to latter update times. The Multi-model does not suffer by this drawback, as it creates a unique model for every time of update.

Table 4.1: Comparison of the benchmark (static) models with the proposed regular updated models using prediction error measures.

Methods	**wMAE**	**MAE_0**
Mean-static	2.814	2.814
Mean-updated ($\delta = 1$)	2.346	2.814
Static-model	2.279	2.279
Single-model ($\delta = 1$)	1.869	2.387
Multi-model ($\delta = 1$)	1.729	2.279

In Table 4.2, we study the role of the frequency of updates. For Multi-model, the higher frequency increases the accuracy of predictions. However, the improvements are relatively minor, suggesting that a few updates are sufficient. It should also be noted that the higher frequency requires more models and data and thus demands more computational and communication resources.

In Figure 4.1 panel A, we compare the prediction accuracy of models as a function of the time offset t^i since the begging of charging sessions. Due to how we processed the data, the maximum connection duration is 24 hours. Therefore, prediction errors significantly decrease

Table 4.2: Influence of update frequency on wMAE for the updated models.

Models	$\delta = 2$	$\delta = 1$	$\delta = 0.5$	$\delta = 0.25$
Mean-updated	2.337	2.346	2.366	2.383
Single-model	1.862	1.869	1.871	1.876
Multi-model	1.752	1.729	1.719	1.714

when the time offset exceeds 16 hours. For better visualisation, we cut off the times of update higher than 18 hours in Figure 4.1 panel A. Again, as expected, the naive model (Mean-updated) displays significantly worse performance than advanced models (Single-model and Mutli-model).However, the overall pattern is similar. Initially, the prediction error grows and reaches the maximum value around $t^i = 2$ hours. As it can be seen in Figure 2.2, a vast majority of sessions are shorter than four hours. Hence, for small values of t^i a large number of sessions contribute to the error. When short sessions terminate, the error decreases and it starts to grow again from $t^i = 8$ hours. For larger values of t^i, the difference in performance between the Single-model and Mutli-model disappears, indicating that Single-model could be sufficient.

Figure 4.1 panel A, we observed an increase of the prediction error for small and intermediate values of t^i. We divided the sessions into overlapping subsets based on their duration to analyse the prediction error. Figure 4.1 panel B, we analysed the values of MAE_{t^i} separately for sessions taking longer than 1, 2, 6 and 10 hours. On average, shorter sessions reach lower MAE_{t^i} values. The error decreases as far as the time of update is lower than the minimal time of session duration in the subset, i.e. time in condition defining subset. The rises of error, also seen in Figure 4.1 panel A, are caused by the fact, that sessions with longer duration are harder to predict. Only such sessions are left when the time offset t^i is larger.

4.2 Irregular updates

In Table 4.3 we compare the performance of regular and irregular updates when using the Multi-model. With the same number of models, irregular updates achieve higher accuracy of

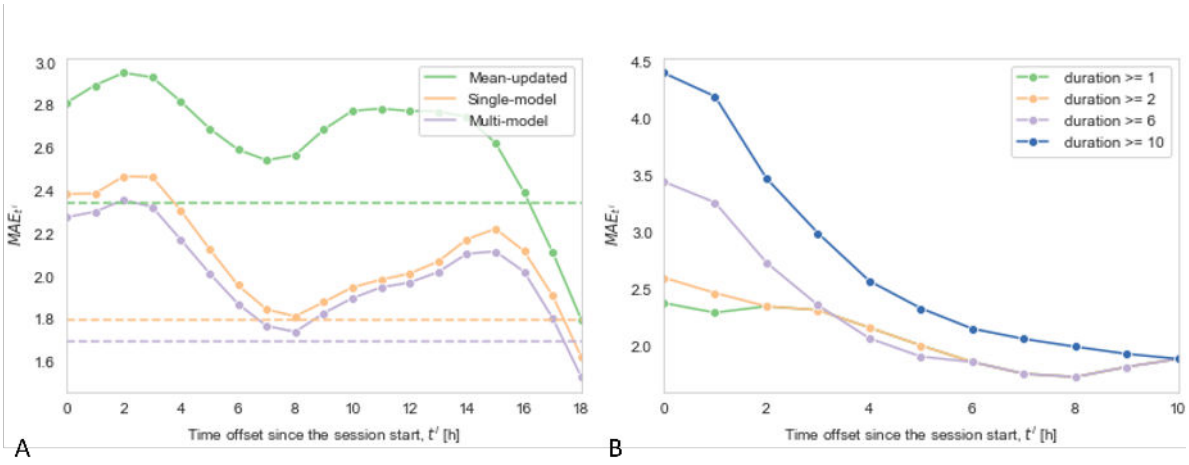

Figure 4.1: Panel A: MAE_{t^i} for different models, $\delta = 1$. Straight horizontal lines show the wMAE value for every method. Panel B: MAE_{t^i} for sessions lasting more than 1, 2, 6 and 10 hours, for the regular Multi-model with $\delta = 1$.

predictions. By comparing the results with Table 4.2, we observe that the regular Multi-model with 96 models ($\delta = 0.25$) gives a similar level of accuracy as irregular Multi-model just with $M = 6$ models.

The obtained time offsets t^i for the irregular Multi-models from Table 4.3 are presented in Figure 4.2. Decline of the active sessions with rising times of update (see Figure 2.2) causes, that frequency of predictions' update is higher for lower times of update. Later on, the updates are approximately uniformly distributed. Since all times of update for multimodel with 12 models are more uniformly distributed and the improvement in Table 4.3 is more significant for lower count of models, we assume that the improvement declines with increasing count of the models.

4.3 Parameters tuning

To reach the best performance, we tune the hyperparameters for LGBM. Since we retrain the model multiple times, we consider just the most relevant models from previous experiments. We evaluate on rolling predictors origin using 3-fold validation with 50 iterations for Bayesian

Table 4.3: Comparison of the regular and irregular strategies.

	Multi-model regular	**Multi-model irregular**
wMAE (M=4)	1.887	1.796
wMAE (M=6)	1.812	1.713
wMAE (M=8)	1.778	1.728
wMAE (M=12)	1.752	1.704

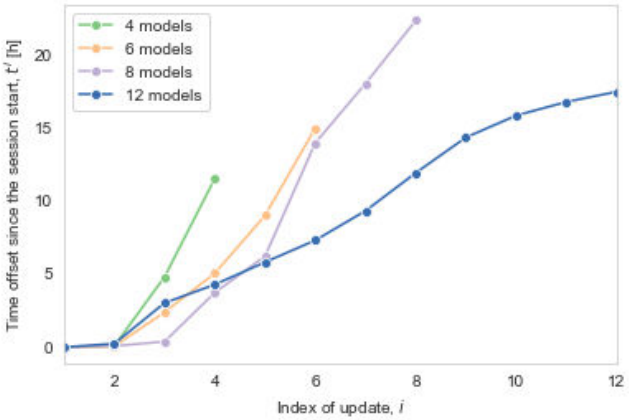

Figure 4.2: Irregular update times for 4, 6, 8 and 12 models.

optimisation, according to Subsection 3.1.3. The following hyperparameters from allowed range were tuned:

- number of leaves: from 100 to 300 with granularity = 10,
- minimal data count in leaves: from 10 to 40 with granularity = 1,
- number of iterations: from 100 to 200 with granularity = 10,
- learning rate: from 0.05 to 0.2 with granularity = 0.01,
- maximal bin count: from 200 to 300 with granularity = 5.

As expected, predictions improve for tuned models. The improvement is higher for multi-model as we tune each model separately, resulting in better performance in summary. More-

Table 4.4: Comparison of models using default hyperparameters for LGBM and tuned models.

Model	**wMAE not tuned**	**wMAE tuned**
Static-model	2.279	2.245
Single-model ($\delta = 4$)	1.898	1.849
Multi-model regular ($\delta = 4$)	1.812	1.765
Multi-model irregular (M=6)	1.713	1.648

over, irregular approach achieves best results after the hyperparameters' optimisation as well as in overall.

4.4 Smart charging evaluation

To assess the usability of the proposed methods and models for smart charging schemes, we observe three main criteria: E^n, E^p and E^o. The goal is to reduce the E^p with minimal E^n. We identified two peak periods with a large energy consumption based on charging patterns from the EVnetNL dataset. The morning peak lasts from 8:00 to 11:00, and the afternoon peak from 17:00 to 21:00. In the following experiment, we set the peak price periods in the same way. We select just the most relevant methods obtained by the previous experiments.

Table 4.5 compares BaU, Optimal and ToU using different prediction models. Optimal approach denotes entire knowledge of the future, thus the charging scheme is optimal in order to real departure of the vehicle. The optimal approach emphasises the potential of smart charging, and it can be used as a benchmark for other approaches. Obviously, optimal charging scheme is superior to all the other models. From ToU schemes, the static-model reaches the best value of energy charged at peak price. However, the demanded but not delivered energy is significantly high. All updating models try to cope with this drawback. As expected, the irregular multi-model performs the best. Surprisingly, the difference in-between the regular and irregular approach is significant. Moreover, the irregular approach decreases the energy charged at peak price besides the not delivered energy, while single-model and regular multi-model decrease merely not delivered energy.

To sum up, the updating models update the prediction in the way that minimises not delivered energy. Additionally, irregular strategy decreases the energy charged in peak period. Hence, it brings the best results in order to three criteria for assessment of the performance for smart charging application.

Table 4.5: The performance of the BaU, Optimal and ToU using different prediction models for charging scheme assessed by three individual criteria (E^n - demanded but not delivered energy, E^p - energy charged at peak price, E^o - energy charged at off-peak price).

Model	$\mathbf{E}^n$ [MWh]	$\mathbf{E}^p$ [MWh]	$\mathbf{E}^o$ [MWh]
BaU	0.0	249.7	369.95
Optimal	0.0	102.25	517.4
Static-model	27.24	96.82	495.59
Single-model ($\delta = 4$)	24.25	106.92	487.91
Multi-model regular ($\delta = 4$)	18.9	118.29	481.93
Multi-model irregular (M=6)	16.81	105.56	493.56

4.5 Charging point availability

Tables 4.6 and 4.7 show the results for prediction of charging point availability in next 15 and 30 minutes respectively. For 30 minutes, the sensitivity, specificity and F1 score are superior to those for 15 minutes. The models reach better specificity than sensitivity. In average, the true negative case is more likely than the true positive case, e.i. models tend to predict unavailability of the charging point truthfully. This trend is more visible for update strategies and may be caused by overestimation of the connection duration in the case of small remaining duration in particular time offset. On the contrary, static model reaches better sensitivity than update models. Thus, the F1 score is similar for all proposed models and the benefit of update of the prediction is not proved.

Note, that we predict connection duration in order to predict categorical target value. Train and predict categorical target value directly can bring different results and impact of the update

can be significant.

Table 4.6: Sensitivity, specificity and precision for prediction, whether session finishes in the next 15 minutes, refers to application for charging point availability.

Model	**Sensitivity**	**Specificity**	**F1 score**
Static-model	0.48	0.78	0.23
Single-model ($\delta = 4$)	0.36	0.91	0.23
Multi-model regular ($\delta = 4$)	0.35	0.92	0.23
Multi-model irregular (M=6)	0.33	0.93	0.24

Table 4.7: Sensitivity, specificity and precision for prediction, whether session finishes in the next 30 minutes, refers to application for charging point availability.

Model	**Sensitivity**	**Specificity**	**F1 score**
Static-model	0.53	0.77	0.37
Single-model ($\delta = 4$)	0.42	0.90	0.35
Multi-model regular ($\delta = 4$)	0.41	0.91	0.36
Multi-model irregular (M=6)	0.40	0.92	0.37

4.6 Features importance

In Table 4.8, we show the most relevant features. Just the first 15 most relevant features are displayed. The LGBM provides importance of the features using the usage for splitting. By dividing that by the total count of splits, we figure out the relative usage of the features. For Multi-models, the average value from all models is used. The last column in Table 4.8 contains average usage from all models.

From the static features both, latitude and longitude of the charging point are important. The majority of the features involve charged energy or statistics of total energy. We

assume that the sessions are more predictable, when we use information from charged energy, maximal power of the charging and historical energy demand from last sessions and users' statistics. Since the charging behaviour of the users follows particular patterns (mentioned in Subsection 2.1.1), important features imply hour of the day, day of the week and month to reflect the macroscopic periodicity in the dataset. Obviously, connection duration and charging duration for user and charging point are relevant.

Table 4.8: First 15 the most relevant features and their relative usage in LGBM splitting for various models.

Feature name	**Single-model** ($\delta = 4$)	**Multi-model** regular ($\delta = 4$)	**Multi-model** irregular (M=6)	**Average**
Hour of the day	17.88%	4.77%	6.08%	9.58%
Mean c.d. for user	5.85%	4.22%	4.46%	4.84%
Day of the week	6.22%	3.09%	3.27%	4.19%
Energy last 10 sessions	4.80%	3.60%	3.60%	4.00%
Connected time last day	3.94%	3.57%	3.60%	3.70%
Mean c.d. for station	3.14%	3.40%	3.45%	3.33%
Energy last session	2.47%	3.64%	3.15%	3.09%
Charged energy	1.89%	2.88%	4.20%	2.99%
Longitude	2.43%	2.98%	2.83%	2.75%
Month	3.48%	2.56%	2.17%	2.74%
Maximal power	2.20%	2.87%	3.03%	2.70%
Energy last 5 sessions	2.10%	3.13%	2.69%	2.64%
Users' relative frequency	2.04%	2.88%	2.78%	2.56%
Mean ch.d. for station	1.80%	3.05%	2.82%	2.56%
Latitude	2.15%	2.83%	2.66%	2.55%

Chapter 5

Conclusion

In this thesis, we explored how the updates of the connection duration predictions can improve the accuracy. We prepared two naive and three advanced prediction models.

From the comparison of their performance, we derived the following conclusions. Regular updates significantly improve the accuracy of predictions. The accuracy of predictions varies when they are done with a different time offset since the beginning of the charging session. This seems to be linked to the way how the properties of charging sessions change with their duration. Irregular updates further improve the accuracy of predictions. Higher frequency of prediction updates is beneficial in the early stages of charging sessions. Later on, approximately uniformly distributed updates are sufficient. Models updating the prediction are more sensitive to hyperparameters' tuning.

Updates of the prediction are beneficial for smart charging. In comparison with BaU charging, proposed models importantly mitigate the energy charged in peak periods and does not significantly increase demanded but not charged energy. As expected, updating models are superior to static-model. for charging point availability, updating models reached better specificity and worse sensitivity, while the F1 score was similar.

On the contrary, the presented research suffers from several limitations, that we will address in future research:

- We used only a single method. By utilising some other methods, e.g., neural networks, some further improvements could be achieved.

- We used only a single dataset. The results might be data-dependent to some degree.
- The prediction updates should also be investigated with other characteristics of charging sessions than the connection duration.
- Since charging point availability is the classification problem, response variable for model training and predicting should be categorical. This can bring different results than we achieved in this thesis.

Bibliography

[1] Valérie Masson-Delmotte et al. The physical science basis. contribution of working group i to the sixth assessment report of the intergovernmental panel on climate change. *IPCC, 2021: Summary for Policymakers*, 2021.

[2] Hauke Engel, Russell Hensley, Stefan Knupfer, and Shivika Sahdev. Charging ahead: Electric-vehicle infrastructure demand. *McKinsey Center for Future Mobility*, page 8, 2018.

[3] Luboš Buzna, Pasquale De Falco, Gabriella Ferruzzi, Shahab Khormali, Daniela Proto, Nazir Refa, Milan Straka, and Gijs van der Poel. An ensemble methodology for hierarchical probabilistic electric vehicle load forecasting at regular charging stations. *Applied Energy*, 283:116337, 2021.

[4] Yu-Wei Chung, Behnam Khaki, Tianyi Li, Chicheng Chu, and Rajit Gadh. Ensemble machine learning-based algorithm for electric vehicle user behavior prediction. *Applied Energy*, 254:113732, 2019.

[5] Oliver Frendo, Nadine Gaertner, and Heiner Stuckenschmidt. Improving smart charging prioritization by predicting electric vehicle departure time. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[6] Kunihiro Miyazaki, Takayuki Uchiba, and Kenji Tanaka. Clustering to predict electric vehicle behaviors using state of charge data. In *2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe*, pages 1–6. IEEE, 2020.

[7] Christoph Goebel and Marcus Voß. Forecasting driving behavior to enable efficient grid integration of plug-in electric vehicles. In *2012 IEEE Online Conference on Green Communications (GreenCom)*, pages 74–79, 2012.

[8] Alexandre Lucas, Ricardo Barranco, and Nazir Refa. Ev idle time estimation on charging infrastructure, comparing supervised machine learning regressions. *Energies*, 12(2):269, 2019.

[9] Haibo He, Sheng Chen, Kang Li, and Xin Xu. Incremental learning from stream data. *IEEE Transactions on Neural Networks*, 22(12):1901–1914, 2011.

[10] Nadeem Ahmed Syed, Huan Liu, and Kah Kay Sung. Incremental learning with support vector machines, 1999.

[11] Aulon Shabani and Orion Zavalani. Hourly prediction of building energy consumption: An incremental ann approach. *European Journal of Engineering and Technology Research*, 2(7):27–32, Jul. 2017.

[12] Nikunj C. Oza and Stuart J. Russell. Online bagging and boosting. In Thomas S. Richardson and Tommi S. Jaakkola, editors, *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics*, volume R3 of *Proceedings of Machine Learning Research*, pages 229–236. PMLR, 04–07 Jan 2001. Reissued by PMLR on 31 March 2021.

[13] H. Grabner and H. Bischof. On-line boosting and vision. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 260–267, 2006.

[14] Aaron Jaech, Baosen Zhang, Mari Ostendorf, and Daniel S. Kirschen. Real-time prediction of the duration of distribution system outages. *IEEE Transactions on Power Systems*, 34(1):773–781, 2019.

[15] Ruimin Li, Francisco C Pereira, and Moshe E Ben-Akiva. Overview of traffic incident duration analysis and prediction. *European transport research review*, 10(2):1–13, 2018.

[16] Francesco Corman and Lingyun Meng. A review of online dynamic models and algorithms for railway traffic management. *IEEE Transactions on Intelligent Transportation Systems*, 16(3):1274–1284, 2015.

[17] Javad Lessan, Liping Fu, and Chao Wen. A hybrid bayesian network model for predicting delays in train operations. *Computers & Industrial Engineering*, 127:1214–1222, 2019.

[18] Pavle Kecman and Rob MP Goverde. Online data-driven adaptive prediction of train event times. *IEEE Transactions on Intelligent Transportation Systems*, 16(1):465–474, 2014.

[19] Great Minster House. Department for Transport. Electric vehicle smart charging. [Online] Available at: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/817107/electric-vehicle-smart-charging.pdf.

[20] Qinglong Wang, Xue Liu, Jian Du, and Fanxin Kong. Smart charging for electric vehicles: A survey from the algorithmic perspective. *IEEE Communications Surveys Tutorials*, 18(2):1500–1517, 2016.

[21] Reza Fachrizal, Mahmoud Shepero, Dennis van der Meer, Joakim Munkhammar, and Joakim Widén. Smart charging of electric vehicles considering photovoltaic power production and electricity consumption: A review. *eTransportation*, 4:100056, 2020.

[22] Chaojie Li, Chen Liu, Ke Deng, Xinghuo Yu, and Tingwen Huang. Data-driven charging strategy of pevs under transformer aging risk. *IEEE Transactions on Control Systems Technology*, 26(4):1386–1399, 2017.

[23] Sangmin Yeo and Deok-Joo Lee. Selecting the optimal charging strategy of electric vehicles using simulation based on users' behavior pattern data. *IEEE Access*, 2021.

[24] Emre C. Kara, Jason S. Macdonald, Douglas Black, Mario Bérges, Gabriela Hug, and Sila Kiliccote. Estimating the benefits of electric vehicle smart charging at non-residential locations: A data-driven approach. *Applied Energy*, 155:515–525, 2015.

[25] Christoph Heilmann and David Wozabal. How much smart charging is smart? *Applied Energy*, 291:116813, 2021.

[26] Nasrin Sadeghianpourhamami, Nazir Refa, Matthias Strobbe, and Chris Develder. Quantitive analysis of electric vehicle flexibility: A data-driven approach. *International Journal of Electrical Power & Energy Systems*, 95:451–462, 2018.

[27] GreenFlux. Ev smart charging whitepaper, 2020. [Online] Available as: `https://www.greenflux.com/whitepaper/`.

[28] Marisca Zweistra, Stan Janssen, and Frank Geerts. Large scale smart charging of electric vehicles in practice. *Energies*, 13(2):298, 2020.

[29] Christian Will and Alexander Schuller. Understanding user acceptance factors of electric vehicle smart charging. *Transportation Research Part C: Emerging Technologies*, 71:198–214, 2016.

[30] Economist. A lack of chargers could stall the electric-vehicle revolution. [Online] Available at: https://www.economist.com/business/a-lack-of-chargers-could-stall-the-electric-vehicle-revolution/21806663.

[31] Paul Sawers. Google maps will now show real-time availability of electric vehicle charging stations, 2019. [Online] Available at: https://venturebeat.com/2019/04/23/google-maps-will-now-show-real-time-availability-of-charging-stations-for-electric-cars/.

[32] Tai-Yu Ma and Sébastien Faye. Multistep electric vehicle charging station occupancy prediction using hybrid lstm neural networks. *Energy*, 244:123217, 2022.

[33] Can Bikcora, Nazir Refa, Lennart Verheijen, and Siep Weiland. Prediction of availability and charging rate at charging stations for electric vehicles. In *2016 International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pages 1–6, 2016.

[34] Christopher Hecht, Jan Figgener, and Dirk Uwe Sauer. Predicting electric vehicle charging station availability using ensemble machine learning. *Energies*, 14(23):7834, 2021.

[35] Francesca Soldan, Enea Bionda, Giuseppe Mauri, and Silvia Celaschi. Short-term forecast of ev charging stations occupancy probability using big data streaming analysis. *arXiv preprint arXiv:2104.12503*, 2021.

[36] Deny Ratna Yuniartha, Nur Aini Masruroh, and Muhammad Kusumawan Herliansyah. An evaluation of a simple model for predicting surgery duration using a set of surgical procedure parameters. *Informatics in Medicine Unlocked*, 25:100633, 2021.

[37] Bindu Vekaria, Christopher Overton, Arkadiusz Wiśniowski, Shazaad Ahmad, Andrea Aparicio-Castro, Jacob Curran-Sebastian, Jane Eddleston, Neil A Hanley, Thomas House, Jihye Kim, et al. Hospital length of stay for covid-19 patients: Data-driven methods for forward planning. *BMC Infectious Diseases*, 21(1):1–15, 2021.

[38] Netherlands Enterprise Agency. Electric vehicle charging — definitions and explanation, 2019. [Online] Available at: https://www.nklnederland.nl/uploads/files/Electric_Vehicle_Charging_-_Definitions_and_Explanation_-_january_2019.pdf.

[39] Milan Straka, Rui Carvalho, Gijs Van Der Poel, and L'uboš Buzna. Analysis of energy consumption at slow charging infrastructure for electric vehicles. *IEEE Access*, 9:53885–53901, 2021.

[40] ElaadNL. [Online] Available at: https://www.elaad.nl/.

[41] Ranjit R Desai, Roger B Chen, and William Armington. A pattern analysis of daily electric vehicle charging profiles: operational efficiency and environmental impacts. *Journal of Advanced Transportation*, 2018, 2018.

[42] Milan Straka and Lubos Buzna. Clustering algorithms applied to usage related segments of electric vehicle charging stations. *Transportation Research Procedia*, 40:1576–1582, 01 2019.

[43] Milan Straka, Lucia Piatriková, Peter van Bokhoven, and L'uboš Buzna. A matrix approach to detect temporal behavioral patterns at electric vehicle charging stations. *Trans-*

portation Research Procedia, 55:1353–1360, 2021. 14th International scientific conference on sustainable, modern and safe transport.

[44] Scikit-learn. Scikit-learn 1.0 documentation - density estimation. [Online] Available at: https://scikit-learn.org/stable/modules/density.html.

[45] Pandas. Function provides rolling window calculations. [Online] Available at: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rolling.html.

[46] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[47] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[48] Scikit-learn and Microsoft. Scikit-learn api for lightgbm. [Online] Available at: https://lightgbm.readthedocs.io/en/latest/Python-API.html#scikit-learn-api.

[49] Scikit-learn. Scikit-learn 1.0 documentation - advanced topics, categorical feature support. [Online] Available at: https://lightgbm.readthedocs.io/en/latest/Advanced-Topics.html.

[50] Scikit-learn. Scikit-learn 1.0 documentation - optimal split for categorical features. [Online] Available at: https://lightgbm.readthedocs.io/en/latest/Features.html#optimal-split-for-categorical-features.

[51] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019.

[52] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[53] Rob J Hyndman. Cross-validation for time series, 2016. [Online] Available at: https://robjhyndman.com/hyndsight/tscv/.

[54] Scikit-learn. Time series cross-validator. [Online] Available at: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.TimeSeriesSplit.html.

[55] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 2009.

[56] Jason Brownlee. What is the difference between test and validation datasets?, 2020. [Online] Available at: https://machinelearningmastery.com/difference-test-validation-datasets/.

[57] Dimitrios Effrosynidis and Avi Arampatzis. An evaluation of feature selection methods for environmental data. *Ecological Informatics*, 61:101224, 2021.

[58] Microsoft. Lightgbm parameters tuning. [Online] Available at: https://lightgbm.readthedocs.io/en/latest/Parameters-Tuning.html.

[59] Sergio Cleger-Tamayo, Juan M Fernandez-Luna, and Juan F Huete. On the use of weighted mean absolute error in recommender systems. *Citeseer*, pages 24–26, 2012.

[60] Robert van den Hoed, Simone Maase, Jurjen Helmus, Rick Wolbertus, Youssef el Bouhassani, Jan Dam, Milan Tamis, and Bronia Jablonska. *E-mobility: getting smart with data*. Hogeschool van Amsterdam, 2019.

[61] Dale Hall and Nic Lutsey. Literature review on power utility best practices regarding electric vehicles. Technical report, International Council On Clean Transportation, 2017. Available as: `https://theicct.org/sites/default/files/publications/Power-utility-best-practices-EVs_white-paper_14022017_vF.pdf`.

[62] Vaibhav Jayaswal. Performance metrics: Confusion matrix, precision, recall, and f1 score, 2020. [Online] Available at: https://towardsdatascience.com/performance-metrics-confusion-matrix-precision-recall-and-f1-score-a8fe076a2262.

List of Attachments

Attachment A: Contents of the attached CD

Attachments

Attachment A: The attached CD consists of:

- Diploma thesis in pdf format.
- Implemented classes and functions in Python.
- Jupyter files performing the experiments and plotting.