UNIVERSITÀ
DI TRENTO

Department of Information Engineering and Computer Science

Master's Degree in
Artificial Intelligence Systems

Final Dissertation

A forecasting system for electricity production and customer demand

Supervisors

Elisa Ricci

Daniele Miorandi

Student

Samuel Bortolin

Academic year 2021/2022

Acknowledgements

Thanks to ...

Contents

Abstract

The abstract is a short summary of the work describing the target, the subject of the thesis, the methodology and the techniques, the data collection and elaboration, the explanation of the reached results and the conclusion. The abstract of the dissertation must have a maximum length of 3 pages and must include the following information:

- context and motivation
- short summary of the main problem you have dealt with
- developed and / or used techniques
- reached results, the personal contribution of the student has to be highlighted

Note: Please note that the approximate number of pages is 70. These 70 pages include:

- table of contents
- abstract
- chapters

Exclude:

- frontispiece (title page)
- acknowledgements
- bibliography
- attachments

1 Introduction

Brief introduction to the work ...

1.1 Problem statement

This is the problem ...

1.2 Approach to the problem

This is the approach ...

1.3 Outline

Here it is written how the thesis is organized ...

2 State of the Art

In this chapter, the current state of the art is analyzed in the context of electricity data representation and time series forecasting methods. In the first section, a brief introduction to the proposed standards for electricity data representation is presented. Subsequently, various technologies presented in the literature for time series forecasting are discussed. In particular, several implementations and use cases are presented. Also, an in-depth discussion on two hot topics, Transformers and Automated Machine Learning (AutoML), is treated in dedicated subsections. Furthermore, the three use cases of interest, electricity demand forecasting, consumption baseline forecasting, and electricity production forecasting, are treated more in detail in dedicated sections. At the end of this chapter, it will be clear the context around which the proposed system is developed.

2.1 Electricity data representation

In this section, a discussion of electricity data representation and proposed standards is presented.

In [21], Chen et al. studied the data quality of electricity consumption data in a smart grid environment. The definition and classification of data quality issues are explained. The data quality issues related to electricity consumption are classified into three types: noisy data, incomplete data, and outlier data. These three types of data quality issues are discussed. The paper introduced the causes of electricity consumption outlier data and provided a review of the possible detection methods. This is a relevant study since most industrial studies in this field use real-world data that presents these issues.

The Green Button Data[1] is an industry initiative in response to the 2012 White House call-to-action to provide customers easy and secure access to their energy usage information in a both consumer-friendly and computer-friendly format. This data may include electricity, natural gas, and water usage. Customers using this service are able to securely download their own detailed energy usage in a standard format with a simple click. They may choose to upload their own data to a third-party application or automate the secure transfer of their own energy usage data to authorized third parties, based on affirmative customer consent and control. It is a very nice initiative in the U.S.and services like UtilityAPI are compatible with the Green Button standard providing support for APIs and XML schemas[2].

[81] provided a case study and explained the lessons learned through the roll-out of Green Button electricity, natural gas, and water data-access initiative, in order to make readily available energy and water consumption data for consumers and third-party companies, that can assist customers while ensuring security and privacy of their data. This paper presented a case study using the Green Button standard and the steps taken to ensure data security and privacy while enabling access to those consumption data by the consumer and third parties. Data security and privacy were achieved through the use of the Green Button standard and subsequent implementation by the Green Button Alliance of a compliance-testing program. Considerations and solutions were needed for data in transit, data at rest, and the authorization mechanisms for allowing unregulated third-party companies to interface directly with utilities on behalf of the consumer while ensuring the consumer maintains complete control of what is to be shared and the ability to revoke that sharing at any time.

[23] presented a big-data-based framework for dealing with electricity consumption behavior. It conducted an analysis of the current state-of-the-art methodologies for the extraction of electro-information. They also analyzed in-depth data for pattern identification, relational analysis, and actions to perform on electricity usage.

[1] `https://www.greenbuttondata.org/`

[2] `https://utilityapi.com/docs/greenbutton`

[116] proposed a novel broker-client system architecture for big data analytics: Smart Meter Analytics Scaled by Hadoop (SMASH). They demonstrated that SMASH is able to perform data storage, query, analysis, and visualization tasks on large data sets at a 20 TB scale. Experimental results suggested that SMASH is able to provide a competitive and easily operable platform to manage big energy data and visualize knowledge, with the potential to provide support to data-intensive decision-making.

2.2 Time series forecasting

In this section, a brief introduction to forecasting competitions, the use of cross-validation for time series predictor evaluation, and techniques for time series forecasting are presented.

Time series forecasting is a classic and well-studied topic. The classical approaches, based on statistical methods such as Auto-Regressive Integrated Moving Average (ARIMA), have been joined in recent years by standard Machine Learning (ML) methods and recently also by Deep Learning (DL) methods with the use of neural networks and transformers.

Forecasting competitions were proposed to promote the development of new solutions and novel techniques. A brief review was proposed in [51]. The first and most influential forecasting competitions were, and currently are, the M-competitions[3]. These competitions promoted the application of the most recent statistical such as ARIMA, ML such as Support Vector Regression (SVR), and DL such as Long Short-Term Memory (LSTM) approaches developed over time to be applied to the forecasting field. Issues like the statistical significance of the results, cheating using part of the test sets, and reproducibility of the results were addressed over the competitions. Also, other competitions were open such as Sante Fe competitions, the KDD cup, Neural network competitions, Kaggle time series competitions, and Global energy forecasting competitions.

[99] discussed whether forecasting competition data are representative of reality. This is a very important point since the performance of new forecasting methods are typically evaluated by exploiting data from past forecasting competitions. However, due to their many limitations, these datasets might not be indicative. Since obtaining a complete picture of the real world is impossible in practice, the paper proposed to use the M4 competition data as an indication of the real world. This is reasonable since this data set is composed of many series from the business world. The properties of this dataset representative of reality are then compared with past datasets, showing that many popular benchmarks may deviate from reality. The main differences observed were referred to the abnormality of the data, in fact, reality data are relatively more skewed series with outliers, as well as their limited randomness/ trend.

In [15], Bergmeir and Benítez presented a study on the use of cross-validation for time series predictor evaluation. They aimed to combine the evaluation of traditional forecasting procedures, on the one hand, and the evaluation of ML techniques on the other hand. In fact, in traditional forecasting, a part from the end of each time series is reserved for testing, and to use the rest is used for training. Instead, when evaluating ML and other regression methods, often cross-validation is used in the evaluation process without paying much attention to the fact that there are theoretical problems with respect to temporal evolutionary effects and dependencies within the data that invalidate the fundamental assumptions of cross-validation. They suggested that the use of a blocked form of cross-validation for time series evaluation should be the standard procedure, thus using all available information and circumventing the theoretical problems. They also affirmed that the use of cross-validation techniques, together with an adequate control for stationarity, led to a more robust model selection.

In [20], Cerqueira et al. studied the application of performance estimation methods to time series forecasting tasks. The dependency among observations in time series raises some caveats about the most appropriate way to estimate performance in this type of data, in fact, cross-validation cannot be applied since requires i.i.d. (independent and identically distributed) data. Results of a comparative study of different performance estimation methods showed noticeable differences among them. In particular, their empirical experiments suggested that blocked cross-validation can be applied to stationary time series. However, when the time series are non-stationary, the most accurate estimates

[3] `https://en.wikipedia.org/wiki/Makridakis_Competitions`

were produced by out-of-sample methods, particularly the holdout approach repeated in multiple testing periods.

The following are relevant studies in the time series forecasting context.

[27] is a paper published in 2006 that review the research into time series forecasting made from 1982 to 2005. Many relevant methods were presented such as exponential smoothing, ARIMA, state space and structural models and the Kalman filter, regime-switching models, functional-coefficient models, neural networks, and many others also involving the combination of approaches. There were also a lot of relevant studies that presented theoretical concepts such as Seasonality, Forecast evaluation and accuracy measures, and Prediction intervals and densities. They concluded by saying that enormous progress has been made in many areas, but they found out that there were a large number of topics that need further development such as multivariate time series forecasting, forecasting methods based on nonlinear models, model selection procedures, robust statistical methods, and improved forecast intervals.

In [3], Nesreen et al. presented a large-scale comparison study for the major ML models for time series forecasting. The models considered were MLP, Bayesian neural networks, radial basis functions, Generalized Regression Neural Networks (GRNN), K-Nearest Neighbor (KNN) regression, Classification and Regression Trees (CART), SVR, and Gaussian processes. The study revealed significant differences between the different methods and proclaimed as the best two methods on the monthly M3 time series competition data the MLP and the Gaussian process regression.

[14] proposed a review and a comparison of different strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. They also considered the effects of deseasonalization, input variable selection, and combination on the strategies. From experimental results, they figured out that: (i) Multiple-Output strategies are the best-performing approaches, (ii) deseasonalization leads to uniformly improved forecast accuracy, and (iii) input selection is more effective when performed in conjunction with deseasonalization.

In [103], Tealab studied the advances in time series forecasting models using Artificial Neural Network (ANN) methodologies. They took into consideration the papers published from 2006 to 2016. From the analysis of the papers, they concluded that, although there are many studies that presented the application of neural network models, few of them proposed new neural network models for forecasting. They found out that these many studies had a similar hybrid methodology that consisted in adjusting a linear time series model, and subsequently using the residuals as the input variables of an ANN model.

[11] proposed a comparative study and analysis of various time series forecasting techniques such as linear regression model, ARIMA, LSTM, and many others. In particular, it explored their limitations and utility for different types of time-series data across different domains.

[90] provided a comprehensive literature review of DL studies with a focus on financial time series forecasting implementation. They categorized the studies according to their intended forecasting implementation areas and grouped them based on their DL model choices, such as Convolutional Neural Networks (CNNs), Deep Belief Networks (DBNs), and LSTM.

In [50], Hewamalage et al. presented an extensive empirical study and an open-source software framework of existing Recurrent Neural Network (RNN) architectures for forecasting. They concluded that RNNs are capable of modeling seasonality directly if the series in the dataset possess homogeneous seasonal patterns; otherwise, they recommend a deseasonalization step. Comparisons against exponential smoothing and ARIMA demonstrated that (semi-) automatic RNN models are not perfect, but they were competitive alternatives in many situations.

In [17], Lim and Zohren analyzed the main architecture used in both one-step-ahead and multi-horizon time-series forecasting. They described how temporal information is incorporated into predictions by each of the models, such as CNNs, RNNs, and networks with attention mechanisms. They highlighted also the recent developments in hybrid DL models, which combine statistical models with neural network components to improve performance.

[74] presented the recent ML advances for time series forecasting. They started analyzing the linear methods, paying more attention to penalized regressions and ensembles of models. They continued presenting nonlinear methods, including tree-based methods, such as Random Forest and Gradient-

Boosted Decision Trees (GBDT), and shallow and deep neural networks, in their feed-forward and recurrent versions. Finally, they also consider ensemble and hybrid models by combining different alternatives.

[124] presented a hybrid methodology that combines ARIMA and ANN models. This was innovative in 2003 and was done in order to take advantage of the unique strengths of ARIMA and ANN models in linear and nonlinear modeling. The experimental results indicated that the combined model can be an effective way to improve the forecasting accuracy achieved by both models used separately.

[19] proposed using multiple Support Vector Machines (SVMs) for time series forecasting. The multiple SVMs that best fit the regions partitioned by a self-organizing feature map are constructed by finding the most appropriate kernel function and the optimal free parameters of SVMs. Simulation results showed that the proposed multiple SVMs achieve significant improvement in the generalization performance in comparison with the single SVMs models. In addition, the multiple SVMs also converge faster and use fewer support vectors.

[119] attempted to develop an automatic ANN modeling scheme for Time-Series Forecasting. This scheme was based on the GRNN, a special type of neural network. By taking advantage of several GRNN properties (i.e., a single design parameter and fast learning) and by incorporating several design strategies (e.g., fusing multiple GRNNs), they were able to make the proposed modeling scheme to be effective for modeling large-scale business time series.

[84] described a new type of ensembles improving the predictive performance with respect to existing ensembles for time series forecasting. In particular, they proposed a new form of diversity generation that explores some specific properties of time series prediction tasks. Their experiments confirmed that the proposed method for generating diversity is able to improve the performance of the equivalent ensembles with standard diversity generation procedures.

In [102], Sean and Letham proposed Prophet, a modular regression model with interpretable parameters that can be intuitively adjusted by analysts with domain knowledge about the time series. They also described a component for measuring and tracking forecast accuracy and flagging forecasts that should be checked manually to help analysts make incremental improvements. This is a crucial component that allows analysts to identify when adjustments need to be made to the model or when an entirely different model may be appropriate.

In [16], Borovykh et al. presented a method for conditional time series forecasting based on an adaptation of the recent deep convolutional WaveNet architecture. The proposed method took advantage of dilated convolutions for considering a broad history horizon when forecasting. Conditioning was performed by applying multiple convolutional filters in parallel to separate time series which allowed fast processing of data and the exploitation of the correlation structure between the multivariate time series. Experimental results showed that the proposed network was well-suited for regression-type problems and was able to effectively learn dependencies in and between the series without the need for long historical time series and tends to outperform linear and recurrent models.

In [64], an evolutionary attention learning approach was introduced to LSTM for multivariate time series prediction. In particular, an evolutionary computation-inspired competitive random search method was proposed to configure in an optimal way the parameters in the attention layer. Experimental results illustrated that the proposed model can achieve competitive prediction performance compared with other baseline methods.

[28] proposed a hybrid system that searches for a suitable function to combine the forecasts of linear and nonlinear models. The proposed system performed linear and nonlinear modeling of the time series and a data-driven combination that searches for the most suitable function, between linear and nonlinear formalisms, and also the number of models that maximizes the performance of the combination. As the linear model, the ARIMA model is used and as nonlinear models, Multi-Layer Perceptron (MLP) and SVR were used. Experimental results showed that the proposed hybrid system attains superior performance when compared with single and hybrid models in the literature.

In [92], Shen et al. proposed a novel time series forecasting model, named SeriesNet, which can fully learn features of time series data in different interval lengths. It is composed of a LSTM network and a dilated causal convolution network. The fact that the proposed model could learn multi-range and multi-level features from time series data led to a higher predictive accuracy compared to those

models using fixed time intervals.

[95] presented the winning submission of the M4 forecasting competition. The submission used a dynamic computational graph neural network system that enabled a standard exponential smoothing model to be mixed with advanced LSTM networks into a common framework. The result was a hybrid and hierarchical forecasting method able to perform better than all the other submitted models.

In the context of building better pricing modeling and forecasting frameworks to meet difficulties, [100] proposed to combine seasonal and trend decomposition utilizing LOESS (locally estimated scatterplot smoothing) and Prophet methodologies to perform a more accurate and resilient time series analysis of Italian electricity spot prices. The proposed method could assist in enhancing projections and providing a better understanding of the variables driving the data. Experimental results showed that the combination of approaches improved the forecast accuracy and lowered the Mean Absolute Percentage Error (MAPE) performance metric by 18% compared to the Prophet baseline model.

2.2.1 Attention and transformers

In this subsection, an overview of attention-based and transformer approaches is presented, with a focus on time series forecasting applications.

[83] provided an overview of the state-of-the-art attention models proposed. To provide a general understanding of attention mechanisms, they defined a unified model suitable for most attention structures. They classified existing attention models according to the following criteria: the softness of attention, forms of input features, input representation, and output representation. They also summarized the network architectures used in conjunction with the attention mechanism and described some applications where attention mechanisms are used to improve performance. Finally, we discuss the interpretability that attention mechanisms provide to the DL models, but also the challenges and prospects of attention models.

[47] presented a review of the developed attention mechanism with a focus on the neural machine translation task. They covered the most adopted attention models and their variants, such as self-attention, soft attention, hard attention, local attention, global attention, additive attention, and multiplicative attention.

[85] analyzed that attention mechanisms have raised significant interest in the research community since they promise relevant improvements in the performance of neural network architectures. In particular, since self-attention was proposed, it has been widely used in transformer-like architectures and has led to significant breakthroughs in many applications. In the work, Pedro and Oliveira performed an objective comparison of a number of different attention mechanisms for the classification of samples in the Skin Cancer MNIST dataset. The results showed that attention modules only sometimes improved the performance of CNN architectures, but also that this improvement was not consistent in different settings. On the other hand, the results obtained with self-attention mechanisms showed consistent and significant improvements, leading to the best results even in architectures with a reduced number of parameters.

In [93], Shih et al. analyzed that the standard attention mechanisms in multivariate time series forecasting only review the information at each previous time step and select relevant information to help generate the outputs. For this reason, they failed to capture temporal patterns across multiple time steps. In the study, they proposed a set of filters to extract time-invariant temporal patterns and then a novel attention mechanism to select relevant time series and use frequency domain information for multivariate time series forecasting. Finally, they applied the proposed approach to many real-world tasks demonstrating that it was able to achieve state-of-the-art performance in most of them.

In [35], Du et al. proposed a novel temporal attention encoder-decoder model to successfully deal with multivariate time series forecasting. They developed an end-to-end DL structure that integrates the traditional idea of the encoder-decoder learning structure and a temporal attention mechanism for jointly learning long-term temporal dependency and hidden non-linear correlation features of multivariate temporal data. Experimental results on five multivariate time series datasets showed that the proposed model had the best forecasting performance compared with baseline models, such as SVR, RNN, CNN, LSTM, and Gated Recurrent Unit (GRU).

[49] aimed to predict the energy use of solar-assisted water heating systems using a novel ML

approach. They proposed to use a LSTM network enhanced by an attention mechanism and the decomposition of input data into sub-layers. They compared the performance of the proposed approach with a feed-forward neural network, a LSTM network, and an Attention-based LSTM neural network. Experimental results showed that the proposed model had superior performance over the conventional models in this task.

Liu et al. analyzed that the current attention-based recurrent neural networks can effectively represent and learn the dynamic spatiotemporal relationships between exogenous series and target series, but they only perform well in one-step time prediction and short-term time prediction. In [68], they proposed dual-stage two-phase-based RNN (DSTP-RNN) for long-term time series prediction. The DSTP-based structure enhanced the spatial correlations between exogenous series. The first phase produces violent but decentralized response weights, while the second phase leads to stationary and concentrated response weights. Then, they employed multiple head attention on target series to boost the long-term dependency. Experimental results demonstrated that the proposed model can be successfully used to develop systems for a wide range of applications, with performance superior to nine baseline methods on four datasets in the fields of energy, finance, environment, and medicine.

Qi et al. analyzed that with the development of attention mechanisms, the frequency and time attention information of audio can be fully exploited, and the amplitude properties of audio can also be better integrated with a good fusion module. In [88], they improved existing frequency-temporal attention by extracting the attention information with the frequency-temporal attention and performing an additive fusion of features. Then, they applied attentional feature fusion based on multi-scale channel attention, and finally, temporal dependencies are learned through a self-attention module. Experimental results on four datasets demonstrated that the proposed model outperformed existing state-of-the-art models.

In the context of Aspect-Based Sentiment Analysis, Feng et al. argued that the original attention mechanism is not the ideal configuration, as most of the time only a small portion of terms are strongly related to the sentiment polarity of an aspect or entity. In fact, standard attention mechanisms apply the complete computed attention weights and do not place any restrictions on the attention assignment. In [39], they proposed a masked attention mechanism with two different approaches to generate the mask. The first approach was done by setting an attention weight threshold that is determined by the maximum of all weights and keeping only attention scores above the threshold. The second approach was done by selecting the top inputs with the highest weights. Both approaches removed the lower score parts that were assumed to be less relevant to the aspect of focus. By ignoring part of the input that is claimed irrelevant, a large proportion of input noise is removed, keeping the downstream model more focused and reducing calculation cost. Experiments showed significant improvements over state-of-the-art models with full attention, demonstrating the effectiveness of the masked attention mechanism.

In [108], Vaswani et al. proposed the Transformer architecture, based solely on attention mechanisms and fully connected feed-forward networks, plus layer normalization and residual connections. Experimental results on machine translation tasks showed that this groundbreaking model was superior in quality achieving better results than existing methods while being more parallelizable and requiring significantly less time to train. Moreover, they also showed that the Transformer is able to generalize to other tasks like English constituency parsing both with large and limited training data.

In [118], Wu et al. presented a new approach to time series forecasting by developing a novel method that employs Transformer-based ML models to forecast time series data. The approach leveraged self-attention mechanisms to learn complex patterns and dynamics from time series data. They created a generic framework that can be applied to univariate and multivariate time series data, as well as time series embeddings. The influenza-like illness (ILI) forecasting was used as a case study. They showed the effectiveness of the proposed approach and that the produced forecasting results are favorably comparable to the state-of-the-art.

[127] discussed the several severe issues of Transformer architecture that prevent it from being directly applicable to long sequence time-series forecasting (LSTF), including quadratic time complexity, high memory usage, and the limitation of the encoder-decoder architecture. To address these issues, Zhou et al. designed an efficient transformer-based model for LSTF, named Informer. It had some

distinctive characteristics, such as a ProbSparse self-attention mechanism, which achieves O(L log L) in time complexity and memory usage, the self-attention distilling highlights dominating attention by halving cascading layer input, the generative style decoder predicts the long time-series sequences in one forward operation rather than a step-by-step way. Extensive experiments on four large-scale datasets demonstrated that Informer significantly outperformed existing methods and provided a new solution to the LSTF problem.

[46] analyzed the state-of-the-art models for Multivariate Time Series Forecasting. The sequence-to-sequence models rely on attention between timesteps, which allows for temporal learning but fails to consider distinct spatial relationships between variables. In contrast, graph neural network methods explicitly model variable relationships, however, often rely on predefined graphs and perform separate spatial and temporal updates without establishing direct connections between each variable at every timestep. The study addressed the presented problems by using a Transformer and translating the multi-variate forecasting task into a spatiotemporal sequence formulation so that each Transformer input represented the value of a single variable at a given time. Using this formulation, Long-Range Transformers can then jointly learn interactions between space, time, and value information. The proposed method, called Spacetimeformer, achieved competitive results on benchmarks of different domains while learning fully-connected spatiotemporal relationships purely from data.

In [66], Lim et al. introduced the Temporal Fusion Transformer (TFT): a novel attention-based architecture that combines high-performance multi-horizon forecasting with interpretable insights into temporal dynamics. The presented TFT relied on recurrent layers for local processing and on interpretable self-attention layers for long-term dependencies. It was able to select relevant features and suppress unnecessary components through a series of gating layers. Experimental results on a variety of real-world datasets demonstrated significant performance improvements over existing benchmarks and they also highlighted some practical interpretability use cases enabled by the TFT architecture.

TFT architecture was a breakthrough in time series forecasting and achieved state-of-the-art performance in several tasks. Hereafter, some studies using this architecture are presented.

In [86], the TFT was applied to forecast the trajectory of future vital signs based on time-varying measurements of past vital signs. This is quite important since the deterioration of a patient's condition is usually preceded by several hours of abnormal physiology as indicated by the patient's vital signs. The model was developed using the Songklanagarind critical care dataset, which includes vital sign measurements from 140 patients. Experimental results showed that TFT was able to capture the temporal dynamics of vital signs and can potentially be used to detect irregular patterns in vital sign time series.

In [125], a TFT was adopted to predict freeway speed with prediction horizons from 5 to 150 minutes. A traffic speed data set was used to train and test the prediction model for demonstrating the advantage provided by TFT. The TFT prediction performance was compared with several classic traffic speed prediction methods, and the results revealed that the TFT performed better than the other classic models when the prediction horizon is longer than 30 minutes. Moreover, the TFT is also more stable when the prediction horizon is 60 minutes or longer.

[117] used a novel forecasting approach for interpretable wind speed prediction by incorporating variational mode decomposition, a TFT model, and an evolutionary algorithm. In the proposed approach, variational mode decomposition was employed to break down the raw wind speed sequence into a set of intrinsic mode functions. Adaptive differential evolution was then used for optimizing several parameters of a TFT allowing it to achieve satisfactory forecasting performance. Empirical studies using real-world wind speed data sets demonstrated that the proposed model outperformed other comparable models in nearly all performance metrics. Moreover, TFT allowed gaining information about the importance ranking of the decomposed wind speed sub-sequences, meteorological data, and attention analysis of different step lengths.

2.2.2 Automated machine learning

In this subsection, an overview of the developed approaches in the AutoML field is presented, with a focus on time series forecasting applications.

[37] exposed the need for automating the process of building good ML models, mainly due to the

fact that with the continuous and vast increase of the amount of data, it was recognized that the number of data scientists cannot be scaled to tackle all this data. In this study, Elshawi et al. presented a comprehensive survey of the state-of-the-art efforts for dealing with the Combined Algorithm Selection and Hyper-parameter tuning (CASH) problem in the ML domain. In addition, they highlighted the research work of automating also the other steps of the full complex ML pipeline from data understanding to model deployment. Furthermore, they provided an overview of the various tools and frameworks that have been proposed. Finally, they discussed some of the research directions and open challenges that need to be addressed in order to achieve the vision and goals of the AutoML process, such as scalability, optimization techniques to use, time budget management, and data preparation.

In [105], Truong et al. investigated the current state of AutoML tools aiming to automate repetitive tasks in ML pipelines, such as data pre-processing, feature engineering, model selection, hyperparameter optimization, and prediction result analysis. They conducted various evaluations of the tools on many datasets to examine their performance and compare their advantages and disadvantages. They observed that most AutoML tools obtain reasonable results in terms of their performance across many datasets and no tool managed to outperform all others on all the tasks. Across the various evaluations and benchmarks they tested, H2O AutoML, Auto-Keras, and Auto-Sklearn performed better than Ludwig, Darwin, TPOT, and Auto-ml.

[79] reviewed the various AutoML, hyperparameter tuning, and meta-learning approaches available and pointed out that most of them are neither properly documented nor very clear due to the differences in the approaches. The strengths and drawbacks of the various approaches and their reviews in terms of algorithms supported, features, and implementations are explored.

[48] presented a comprehensive review of the state-of-the-art in AutoML. According to the DL pipeline, He et al. introduced AutoML methods covering data preparation, feature engineering, hyperparameter optimization, and Neural Architecture Search (NAS). They summarized the representative NAS algorithms' performance on the CIFAR-10 and ImageNet datasets and discussed relevant topics of the NAS methods such as one/two-stage NAS, one-shot NAS, joint hyperparameter and architecture optimization, and resource-aware NAS. Finally, they discussed some open problems related to the existing AutoML methods for future research such as flexible search space, areas exploration, interpretability, reproducibility, and robustness.

[32] observed that data scientists cannot tackle the growing number of challenging tasks due to a lack of expertise and experience across all task domains. To help address this issue, the study provided a survey of ongoing research in the field of Meta-Learning and AutoML. It covered AutoML tools such as TransmogrifAI, Auto-Sklearn, AutoGluon, and NNI. It also summarized possible uses of Meta-Learning for DL, few-shot learning, and automating the ML process.

In [57], Karmaker et al. introduced a new classification system for AutoML systems, using a tier schematic to distinguish systems based on their level of autonomy. They described what an end-to-end ML pipeline actually looks like and analyzed which subtasks have been automated and which are done manually. In fact, most AutoML systems require human involvement in some steps, including understanding the attributes of domain-specific data, defining prediction problems, creating a suitable training dataset, and selecting a promising ML technique. These steps often require a prolonged back-and-forth that makes this process inefficient and keeps AutoML systems from being truly automatic. Next, they introduced the proposed level-based taxonomy for AutoML systems and define each level according to the scope of automation support provided. Finally, they discussed the challenges that stand in the way of automating the whole end-to-end ML pipeline.

In [22], Chen et al. described AutoML as a bi-level optimization problem, where one problem is nested within another to search the optimum in the search space, and reviewed the current developments of AutoML in terms of three categories: automated feature engineering (AutoFE), automated model and hyperparameter tuning (AutoMHT), and Automated Deep Learning (AutoDL). The state-of-the-art techniques and frameworks in the three categories are presented. They concluded by presenting the open challenges of AutoML such as the lack of authoritative benchmarks, efficiency, design of search spaces, and interpretability.

In 2015, Feurer et al. stated that to be effective in practice, ML systems need to automatically choose a good algorithm and feature preprocessing steps for a new dataset at hand, and also set their

respective hyperparameters. Starting from existing work on efficient Bayesian optimization methods, in [42] they presented a robust new AutoML system based on the scikit-learn framework: Auto-Sklearn. It was using 15 classifiers, 14 feature preprocessing methods, and 4 data preprocessing methods, giving rise to a structured hypothesis space with 110 hyperparameters. Auto-Sklearn improved on existing AutoML methods by automatically taking into account past performance on similar datasets, and by constructing ensembles from the models evaluated during the optimization. Results on over 100 diverse datasets showed that Auto-Sklearn substantially outperformed the previous state of the art in AutoML.

In [56], Jin et al. proposed a novel framework for efficient NAS enabling Bayesian optimization to guide the network morphism, which essentially keeps the functionality of a neural network while changing its neural architecture. The framework, called Auto-Keras, developed a neural network kernel and a tree-structured optimization algorithm to efficiently explore the search space. Extensive experiments on real-world benchmark datasets demonstrated the superior performance of the developed framework over the state-of-the-art NAS methods.

In [36], Dyrmishi et al. presented a methodology and framework for using Meta-Learning techniques to develop new methods that serve as effective decision support for the AutoML process. Meta-Learning allows learning from previous experience gained during applying various learning algorithms on different types of data and helps reduce the time needed to learn new tasks. In particular, they used Meta-Learning techniques to answer several crucial questions for the AutoML process such as which classifiers are expected to be the best performing on a given dataset, whether it is possible to predict the training time of a classifier, and which classifiers are worth investing a larger portion of the time budget to improve their performance by tuning them. In the Meta-Learning process, they used 200 datasets with different characteristics and 30 classifiers from Weka and Scikit-learn libraries. Results and Meta-Models have been obtained in a fully automated way.

[45] introduced an open-source benchmark framework for comparing different AutoML systems, that follows best practices and avoids common mistakes. The framework is extensible both in terms of AutoML frameworks and tasks. They used the framework to conduct a thorough comparison of 4 AutoML systems (Auto-WEKA, Auto-Sklearn, TPOT, and H2O AutoML) across 39 datasets and analyze the results. The presented results highlighted the need for further AutoML research. In fact, on some datasets, none of the frameworks outperformed a Random Forest within 4 hours, and high-dimensional or highly multi-class problems were often challenging.

In [106], Vaccaro et al. reviewed some ML models and methods proposed in the literature to analyze their strengths and weaknesses. Then, they proposed their use, alone or in combination with other approaches, to provide possible valid AutoML solutions. They analyzed these solutions from a theoretical point of view and evaluated them empirically on three Atari games. The objective of the study was to identify what could be some promising ways to create effective AutoML frameworks able to replace the human expert as much as possible and thereby making easier the process of applying ML approaches to typical problems of specific domains.

Nguyen et al. investigated that most AutoML-based Bayesian optimization approaches convert the AutoML optimization problem into a Hyperparameter Optimization (HPO) problem, i.e., by modeling the choice of algorithms as an additional categorical hyperparameter. They pointed out that using this approach algorithms and their local hyper-parameters are referred to at the same level, and this makes the initial sampling less robust. In [82], they attempted to formulate the AutoML optimization problem as a Bayesian optimization problem instead of transferring it into a HPO problem. They proposed a novel initial sampling approach to maximize the coverage of the AutoML search space to help Bayesian optimization construct a robust surrogate model. They experimented with this approach on 2 independent scenarios of AutoML with 2 operators and 6 operators over 117 benchmark datasets. Experimental results demonstrated that the performance of Bayesian optimization significantly improved by using the proposed sampling approach.

In [41], Feurer et al. introduced a new AutoML approach, named PoSH (Portfolio Successive Halving) Auto-Sklearn, which enabled AutoML systems to work well on large datasets under rigid time limits by using a meta-learning technique and a bandit strategy for budget allocation. They also studied how to let AutoML explore the design space of AutoML and automatically select the

best configuration by itself. These changes combined give rise to the next generation of their original Auto-Sklearn AutoML system, called Auto-Sklearn 2.0. They verified the improvements of these additions in an extensive experimental study on 39 AutoML benchmark datasets and compared the results to other popular AutoML frameworks and Auto-Sklearn 1.0, showing that the relative error is reduced by up to a factor of 4.5, and yielding a performance in 10 minutes that was better than what Auto-Sklearn 1.0 achieved within an hour.

In [129], Zimmer et al. presented Auto-PyTorch, which jointly and robustly optimized the network architecture and the training pipelines with their hyperparameters to enable fully AutoDL. Auto-PyTorch achieved state-of-the-art performance on several tabular benchmarks by combining multi-fidelity optimization with portfolio construction for warm starting and ensembling of Deep Neural Networks (DNNs) and common baselines for tabular data. To verify the effectiveness of the proposed framework, they introduced a new benchmark on learning curves for DNNs and run extensive studies on typical AutoML benchmarks showing that Auto-PyTorch performed better than several state-of-the-art competitors.

[40] presented a benchmark of AutoML tools. Firstly, it analyzed the characteristics of eight recent open-source AutoML tools (Auto-Keras, Auto-PyTorch, Auto-Sklearn, AutoGluon, H2O AutoML, rminer, TPOT, and TransmogrifAI) and described twelve popular OpenML datasets that were used as benchmarks for different tasks such as regression, binary and multi-class classification tasks. Then, it performed an experimental comparison study by considering predictive scores and computational effort. The best predictive results achieved by the AutoML tools were compared with the best OpenML public results showing that AutoML tools obtained competitive results, outperforming the best OpenML models in five datasets. The results confirmed the potential of AutoML tools to fully automate the ML algorithm selection and tuning.

[54] presented a data augmentation method to significantly improve the performance of neural networks since their modeling capacities in time-series forecasting are limited in cases where enough data may not be available. The proposed method, called Augmented-Neural-Network, used forecasts from statistical models which help produce competitive results on intermediate-length time series. They showed that data augmentation, when paired with AutoML techniques such as NAS, can help to find an adequate neural architecture for a given time series. Using this combination, they demonstrated that led to significant enhancement in the forecasting accuracy of some neural network-based models used on a COVID-19 dataset. The improvement in forecasting accuracy with respect to the same neural networks that do not use augmented data was around 20%.

[7] conducted experiments on time series forecasting using ML, DL, and AutoML techniques. The datasets used in the experiments were quantitative data of the real prices of the currently most used cryptocurrencies. The results demonstrated that AutoML for time series is still in the development stage and needs more study to be the main solution to adopt since it was unable to outperform manually designed ML and DL models.

In recent years AutoDL systems efficiency has tremendously improved, but little attention has been paid to AutoDL frameworks for time series forecasting. In [30], Deng et al. proposed an efficient approach for the joint optimization of neural architecture and hyperparameters of the entire data processing pipeline for time series forecasting. In contrast to common NAS search spaces, they designed a novel NAS space covering various state-of-the-art architectures, allowing for an efficient macro-search over different DL approaches. To efficiently search in such a large configuration space, they employed Bayesian optimization with multi-fidelity optimization. They empirically studied several different budget types enabling efficient multi-fidelity optimization on different forecasting datasets. Furthermore, they compared the final system, called Auto-PyTorch-TS, against several established baselines and showed that it significantly outperformed all of them across several datasets.

2.3 Electricity demand forecasting

In this section, the techniques for electricity demand forecasting are presented. Most of the presented techniques are considered very simple nowadays and they rely on large aggregated data, both on the number of people considered (e.g., the consumption generated by an entire country) or on a very large temporal aggregation (up to 1 year aggregated data). Our use case is limited to the customers of a

small company, from 2 to 4 thousand customers, and need to provide forecasts on an hourly basis for a one-month time horizon.

[94] presented a review of electricity demand forecasting techniques. They classified load forecasting into three categories: short-term forecasts which are usually from one hour to one week, medium forecasts which are usually from a week to a year, and long-term forecasts which are longer than a year. Based on the various types of studies presented, load forecasting techniques may be presented in three major groups: Traditional Forecasting techniques (regression methods, exponential smoothing, iterative reweighted least-squares), Modified Traditional Techniques (adaptive demand forecasting, Auto-Regressive (AR), Auto-Regressive Moving Average (ARMA), ARIMA, SVM) and Soft Computing Techniques (Genetic Algorithms (GAs), fuzzy logic, neural networks, knowledge-based expert systems). From the work, it can be inferred that demand forecasting techniques based on soft computing methods are gaining major advantages for their effective use. There is also a clear move towards hybrid methods, which combine two or more of these techniques.

[101] investigated the use of weather ensemble predictions in electricity demand forecasting from 1 to 10 days ahead. They proposed a weather ensemble prediction by considering 51 scenarios for a weather variable. For each scenario, they produced a scenario for the weather-related component of electricity demand. The results showed that the average of the demand scenarios is a more accurate demand forecast than that produced using traditional weather forecasts. The mean of the 51 scenarios is mathematically equivalent to taking the expectation over the weather-related component of the demand probability density function. They also used the distribution of the demand scenarios to estimate the demand forecast uncertainty.

In [76], Mirasgedis et al. presented how to incorporate weather into models for mid-term electricity demand forecasting. They studied the daily and monthly electricity demand. They noticed that the monthly model performs better thanks to the higher level of aggregation but also that the influence of weather on electricity demand is in a more aggregated way and thus may not account well for the influence of unusual or extreme weather on electricity consumption. The temperature of the day in which electricity demand is projected, the temperature of the two previous days, and the relative humidity have been found to be the most important weather parameters that affect electricity consumption in the Greek interconnected power system.

[89] proposed a first use of a neural network with the Backpropagation learning algorithm for Lao state yearly electricity demand forecasting. They compared it with a regression analysis model showing the higher effectiveness of the neural network.

[31] proposed two models for short-term Singapore electricity demand forecasting: the multiplicative decomposition model and the Seasonal ARIMA (SARIMA) model. Results showed that both models can accurately predict the short-term Singapore demand and that the Multiplicative decomposition model slightly outperforms the SARIMA model.

[131] is an empirical study in which some forecasting models are developed for electricity demand using publicly available data and three models based on ML algorithms. The performance of these models is compared by using different evaluation metrics. The data consists of several measurements of the electricity market in Turkey from 2011 to 2016 and is available for different time granularities (from hourly to yearly aggregated). According to the best result of MAPE, electricity demand was predicted with a 1.4 percentage error with the Random Forest model.

In [130], a modeling approach based on association rules was proposed. Association rules are useful to describe a model in terms of cause and effect. It did not outperform ARIMA but helped to locate the most frequent patterns of electricity consumption.

In [5], Al-Musaylh et al. addressed the short-term electricity demand forecasting with MARS (Multivariate Adaptive Regression Spline), SVR, and ARIMA models using aggregated demand data of Queensland, Australia. They found out that the MARS and SVR models can be considered more suitable for short-term electricity demand forecasting when compared to the ARIMA model. As expected, given its linear formulation in the modeling process, the ARIMA model's performance was lower for all forecasting horizons as it generated very high forecast errors. The study found that the MARS models were able to provide a powerful, yet simple and fast forecasting framework when compared to the SVR models.

[72] presented a method based on a SVR to forecast building energy consumption in southern China. To improve the reliability of SVR in building energy consumption prediction, multiple parameters including weather data (such as yearly mean outdoor dry-bulb temperature, relative humidity, and global solar radiation) and economic factors (such as the ratio of urbanization, gross domestic product, household consumption level and total area of the structure) are taken as inputs.

In [104], a RNN-based time series approach for forecasting Turkish electricity load was proposed. RNNs, LSTM networks, and GRU networks are used. The resulting 0.71% MAPE of their experiments yields better results than existing methods based on ARIMA and ANNs on Turkish electricity load forecasting which have 2.6% and 1.8% MAPE respectively.

In [34] a novel hybrid forecasting system was successfully developed. It was composed of four modules: data preprocessing module, optimization module, forecasting module, and evaluation module. In the data preprocessing module, a signal processing approach is employed to decompose, reconstruct, identify, and mine the primary characteristics of the electrical power system time series. Optimization algorithms are also employed to optimize the parameters of these individual models in the optimization and forecasting modules. Experimental results showed that the hybrid system can be able to satisfactorily approximate the actual value. This is an interesting study from which to take inspiration for the system modeling structure, which can be intended for general-purpose and not just for the electrical power use case.

[59] proposed a recurrent inception CNN (RICNN) that combines RNN and 1-dimensional CNN (1-D CNN). They used the 1-D convolution inception module to calibrate the prediction time and the hidden state vector values calculated from nearby time steps. By doing so, the inception module generates an optimized network via the prediction time generated in the RNN and the nearby hidden state vectors. The proposed RICNN model has been verified in terms of the power usage data of three large distribution complexes in South Korea. Experimental results demonstrate that the RICNN model outperforms the benchmarked multi-layer perception, RNN, and 1-D CNN in daily electric load forecasting (48-time steps with an interval of 30 minutes). This is an interesting network structure that can be suited for our specific use case but has to be effective also on a wider time horizon.

In [13], Bedi and Toshniwal proposed a DL-based framework to forecast electricity demand by taking care of long-term historical dependencies. In fact, existing methods are able only of handling short-term dependencies. The proposed approach is called D-FED and is based on a LSTM network and a moving window-based multi-input multi-output mapping approach of active learning. It is applied to the electricity consumption data of Union Territory Chandigarh, India. The performance of the proposed approach is evaluated by comparing the prediction results with ANN, RNN, and SVR models.

In [78], Muzaffar and Afshari have picked up electrical load data with exogenous variables including temperature, humidity, and wind speed, and used them to train a LSTM network. It has been shown in this work that LSTM outperforms the other traditional methods such as ARMA, SARIMA, and ARMA with exogenous inputs (ARMAX) reducing the percentage of errors in forecasting the load time series. From the study, they found out that LSTM can learn the seasonality patterns and the trend as well instead of extracting these features a priori.

In [115], Wen et al. proposed a DL model to forecast the load demand of aggregated residential buildings with a one-hour resolution, while considering its complexity and variability. Hourly-measured residential load data in Austin, Texas, USA were used to demonstrate the effectiveness of the proposed model, and the forecasting error was quantitatively evaluated using several metrics. The used model is a deep RNN model with GRU (DRNN-GRU). This model assumes knowledge of the future weather data to make a forecast, which would affect the accuracy due to the weather uncertainty over a short to medium period. The results showed that the proposed model forecasts the aggregated and disaggregated load demand of residential buildings with higher accuracy compared to conventional methods.

[24] presented a robust short-term electrical load forecasting framework that can capture variations in building operation, regardless of building type and location. Nine different hybrids of recurrent neural networks and clustering are explored. The test cases involve five commercial buildings of five different building types, i.e., academic, research laboratory, office, school, and grocery store.

Load forecasting results indicate that the DL algorithms implemented in the paper deliver 20-45% improvement in load forecasting performance as compared to the current state-of-the-art results for both hour-ahead and 24-ahead load forecasting. It is found that: (i) the use of hybrid DL algorithms can take as less as one month of data to deliver satisfactory hour-ahead load prediction, (ii) 15-minute resolution data, if available, delivers a 30% improvement in hour-ahead load forecasting, and (iii) the formulated methods are found to be robust against weather forecasting errors. This study gives useful insights about the quantity and granularity of data needed for a short-term prediction range, these are great findings but do not generalize to longer prediction ranges.

In [110], Wang et al. proposed a novel approach based on LSTM network for predicting periodic energy consumption. They stated that this is novel since general forecasting methods do not concern periodicity. Hidden features are extracted by the autocorrelation graph among the real industrial data. Experiments using a cooling system under one-step-ahead forecasting are conducted to verify the performance of LSTM. It was compared with several traditional forecasting methods, such as the ARMA model, Auto-Regressive Fractional Integrated Moving Average (ARFIMA) model, and back propagation neural network (BPNN). The Root Mean Square Error (RMSE) of LSTM is 19.7%, 54.85%, and 64.59% lower than BPNN, ARMA, and ARFIMA on the test data. Furthermore, they demonstrated that the proposed algorithm had the highest generalization capability.

In [113], Wang et al. proposed a stacking model capable of combining the advantages of various basic prediction algorithms and transforming them into meta-features to ensure that the final model can observe datasets from different spatial and structural angles. Load data retrieved from two educational buildings in the coastal city of Tianjin, China, is employed for the case study. The case study buildings are made of classrooms for students and offices for university staff. Experimental results indicated that the stacking method achieves better performance than other tested ML models (Random Forest, GBDT, Extreme Gradient Boosting (XGBoost), SVM, and KNN) regarding accuracy, generalization, and robustness.

In [98], Somu et al. presented kCNN-LSTM, a DL framework that operates on the energy consumption data recorded at predefined intervals to provide accurate building energy consumption forecasts. kCNN-LSTM employs: (i) k-means clustering to perform cluster analysis to understand the energy consumption pattern/trend; (ii) CNN to extract complex features with non-linear interactions that affect energy consumption; and (iii) LSTM network to handle long-term dependencies through modeling temporal information in the time series data. The performance of kCNN-LSTM was compared with the k-means variant of the state-of-the-art energy demand forecast models in terms of Mean Square Error (MSE), RMSE, Mean Absolute Error (MAE), and MAPE showing the efficiency of kCNN-LSTM model over other models in providing accurate energy consumption demand forecasting.

In [65], an attention-based DL model with interpretable insights into temporal dynamics is presented to forecast short-term loads. The TFT included a sequence-to-sequence model, which processes the historical and future covariates to enhance the forecasting performance. Gated Residual Network (GRN) is applied to drop out unnecessary information and improve efficiency. The proposed method is tested on anonymized data from a university campus with a time resolution of 30 minutes. The anomalies and missing data (around 8.85% of the total data) are imputed with the KNN method. The testing results demonstrate the effectiveness of the proposed method achieving less than 5% MAPE. This is an interesting work on a TFT model, which should be extended to obtain great performance also with a higher prediction range in order to be applicable to the use case treated in this thesis.

[62] presented a probabilistic forecasting method for hourly load time series based on an improved TFT (ITFT) model to achieve more accurate and thorough forecasting results. Hourly load time series was reconstructed into multiple day-to-day load time series at different hour points. ITFT model replaces the LSTM with a GRU to learn long-term dependence more efficiently. Quantile constraints and prediction interval (PI) penalty terms were incorporated into the original quantile loss function to prevent quantile crossover and construct more compact PIs. The results show that the proposed method is explanatory and can significantly improve the reliability and compactness of probabilistic load forecasting results compared with other popular methods, such as Quantile Regression Neural Network (QRNN) and Temporal Convolutional Network (TCN).

2.4 Consumption baseline forecasting

In this section, the techniques for consumption baseline forecasting are presented. As demonstrated by most of the following papers analyzed, computing forecasts for a single customer is a more complicated use case compared to energy demand forecast over a customer base. Single customer data is more noisy and very few works consider a single habitation, most of them consider commercial buildings, offices, or schools which reduce the level of noise.

The EU research project S3C developed and tested different guidelines and tools, of particular interest is the guideline on how to create a consumption baseline[4]. The baseline is the reference used to assess the effects of the demand response of a given consumer or set of consumers. The demand response effect is defined as the difference between the metered consumption and the baseline calculation. They explained that the baseline calculation method consists of three criteria: i) data selection method, ii) estimation method, and iii) result adjustment. They pointed out that the combination of these criteria depends on user consumption, weather dependency (including seasonal behavior), and load behavior and should all together fit the user consumption pattern.

In [29], Deb et al. presented a comparison of different time series forecasting techniques for building energy consumption: ANN, ARIMA, SVM, Case-Based Reasoning (CBR), Fuzzy time series, Grey prediction model, Moving average and exponential smoothing (MA & ES), KNN prediction method and Hybrid models. Also, hybrid models are reviewed and analyzed, i.e., the combination of two or more forecasting techniques. The various combinations of the hybrid model are found to be the most effective in time series energy forecasting for single buildings.

[8] aimed to compare the prediction capabilities of five different intelligent system techniques in forecasting the electricity consumption of an administration building. These five techniques are; Multiple Regression (MR), Genetic Programming (GP), ANN, DNN, and SVM. The prediction models are developed based on five years of observed data of five different additional parameters such as solar radiation, temperature, wind speed, humidity, and weekday index. The weekday index is demonstrated as an important parameter since makes it possible to differentiate between working and non-working days. ANN performs better than all other four techniques with a MAPE of 6% whereas MR, GP, SVM, and DNN have MAPE of 8.5%, 8.7%, 9%, and 11%, respectively.

Ahmad et al. in [2] focused on reviewing data-driven approaches and large-scale building energy predicting-based approaches. A thorough review of different techniques is presented in the study, including ANN, SVM, clustering-based, statistical, and ML-based approaches.

[38] investigated the performance of different strategies for multi-step ahead building energy predictions. The results of the study demonstrated the potential of recurrent models for short-term building energy predictions. This study provided references for developing advanced DL models for practical applications.

In [69], Lusis et al. studied how calendar effects, forecasting granularity, and the length of the training set affect the accuracy of a day-ahead load forecast for residential customers. They demonstrated that regression trees, neural networks, and SVR yielded similar average RMSE results, but also that statistical analysis showed that the regression trees technique is significantly better. The use of historical load profiles with daily and weekly seasonality, combined with weather data, leaves the explicit calendar effects with very low predictive power. It was also found that one year of historical data is sufficient to develop a load forecast model for residential customers as a further increase in the training dataset has a marginal benefit. In the setting studied in the paper, it was shown that forecast errors can be reduced by using a coarser forecast granularity. That is expected since aggregating over time will reduce the variability of the data, as also demonstrated in this thesis, good results on hourly forecasts will be almost impossible to achieve and instead by using a daily forecast results will be more acceptable.

In [61], Kim et al. examined a number of different data mining techniques and demonstrated GBDT to be an effective method to build baseline electricity usage. They trained GBDT on data prior to the introduction of new pricing schemes and applied the known temperature following the introduction of new pricing schemes to predict electricity usage with the expected temperature correction. Their

[4] https://www.smartgrid-engagement-toolkit.eu/fileadmin/s3ctoolkit/user/guidelines/GUIDELINE_HOW_TO_CREATE_A_CONSUMPTION_BASELINE.pdf

experiments and analyses showed that the baseline models generated by GBDT capture the core characteristics over the two years with the new pricing schemes. In contrast to the majority of regression-based techniques which fail to capture the lag between the peak of daily temperature and the peak of electricity usage, the GBDT-generated baselines are able to correctly capture the delay between the temperature peak and the electricity peak. Furthermore, subtracting this temperature-adjusted baseline from the observed electricity usage, they found that the resulting values are more amenable to interpretation, which demonstrates that the temperature-adjusted baseline is indeed effective. Instead of providing accurate short-term forecasts, their baseline model aims to capture intraday characteristics that persist for years.

In [87], Platon et al. developed predictive models by using ANN and CBR for producing hourly predictions of a building's electricity consumption. CRB is based on the concept that the current trend of the building's electrical use can be approximated using past trends occurring at similar conditions. They showed the supremacy of ANN over CBR in doing the predictions.

In [55], Jie et al. proposed a baseline load forecasting and optimization method based on non-demand-response factors, considering the effects of non-demand-response factors on customer load characteristics and customer baseline load (CBL) forecasting. The proposed method combines non-demand-response factors mining, similar days selecting, and CBL calculating. A combined calculation model is adopted to predict the CBL. The case study reveals the greater accuracy of this method compared to average, linear regression, and neural network methods.

Forecast at the household level is also getting more and more popular in smart building control and demand response program. This popularity inspired Dong et al. to develop in [33] a hybrid model to address the problem of residential hour and day-ahead load forecasting through the integration of data-driven techniques. They evaluated five different ML algorithms: ANN, SVR, least-square SVM (LS-SVM), Gaussian process regression (GPR), and Gaussian mixture model (GMM). They applied these models to four residential data sets obtained from smart meters. A subdivision of air conditioning (AC) consumptions and not-AC was possible and this led to better results with respect to the total consumption. The final results showed that the hybrid model led to improvements compared to the other ML algorithms for both hour-ahead and 24-h ahead predictions.

In [77], Mocanu et al. investigated two newly developed stochastic models for time series prediction of energy consumption, namely Conditional Restricted Boltzmann Machine (CRBM) and Factored Conditional Restricted Boltzmann Machine (FCRBM). The assessment of the two models is made on a benchmark dataset consisting of almost four years of one-minute resolution electric power consumption data collected from an individual residential customer. As the prediction horizon is increasing, FCRBMs and CRBMs seem to be more robust and their prediction error is typically half that of the ANN. In addition from other experiments, it can be observed that all methods perform better when predicting the aggregated active power consumption, than predicting the demand of intermittent appliances (e.g., electric water heater) recorded from sub-meterings.

In [6], a robust ensemble model was proposed to predict day-ahead mean daily electricity consumption on the household level. The proposed ensemble learning strategy utilized a two-stage resampling plan, which generated diversity-controlled but random resamples that were used to train individual ANN members. Experimental results on a case study showed that the proposed ensemble is able to generate better estimates compared to ANN models and the Bagging ensemble.

To counter the high nonlinearity between inputs and outputs of building energy consumption prediction models, in [126] a novel vector field-based SVR method is proposed. Through multi-distortions in the sample data space or high-dimensional feature space mapped by a vector field, the optimal feature space is found, in which the high non-linearity between inputs and outputs is approximated by linearity. A large office building in a coastal town in China is used for a case study, and its summer hourly cooling load data are used as energy consumption data. The proposed method ensures high accuracy, generalization ability, and robustness for building energy consumption prediction.

In [91], Shao et al. studied and analyzed the energy consumption of hotel buildings by developing a SVM energy consumption prediction model. The SVM model took as input variables the weather parameters and operating parameters of the hotel air-conditioning system. They selected as the kernel

function the RBF (Radial Basis Function) kernel function and optimized the parameters of the kernel for finding the best accuracy for the model predictions. The MSE value of the final model prediction in the case study was 2.22% and R^2 (coefficient of determination) was 0.94. This use case is different from the standard single-building forecasts since a hotel includes different rooms and aggregates the consumption over them, then it is more influenced by periods of the year where there could be more or fewer customers that require different levels of demand. Moreover, in this case, hotel parameters like the air conditioning system are available and help in reducing the overall forecast error.

In [114], a probabilistic load forecasting method for individual consumers is proposed to handle the variability and uncertainty of future load profiles. Pinball loss-guided LSTM network is used to model both the long-term and short-term dependencies within the load profiles. Forecasting for both residential and commercial consumers is tested. Experimental results over different customers showed that the proposed method had superior performance over traditional methods such as QRNN, GBDT, and traditional LSTM.

[18] aimed to use DL-based techniques for day-ahead multi-step load forecasting in commercial buildings. The RNN and CNN models have been proposed and formulated under both recursive and direct multi-step manners. The performance is compared with the SARIMA with exogenous inputs (SARIMAX) model. The gated 24-h CNN model, performed in a direct multi-step manner, proves itself to have the best performance, improving the forecasting accuracy by 22.6% compared to that of the SARIMAX, demonstrating the supremacy of DL models in this kind of use cases.

In [60], Kim and Cho proposed a CNN-LSTM neural network for extracting spatial and temporal features to effectively predict housing energy consumption. This is demonstrated to be an effective architecture since the CNN layer can extract the features between several variables affecting energy consumption, and the LSTM layer is able to model temporal information of irregular trends in time series components. The CNN-LSTM method achieved very good prediction performance for housing energy consumption that previously was even difficult to predict. It recorded the smallest value of root mean square error compared to the conventional forecasting methods for the dataset on individual household power consumption. It managed to predict complex electric energy consumption and obtained the highest performance in all cases of minutely, hourly, daily, and weekly unit resolutions compared to other methods. They stated also that having household characteristics such as occupancy and behavior of the residents might have a large influence on predicting electric energy consumption and could improve the performance of the model.

In [97], Somu et al. proposed a hybrid model for building energy consumption forecasting using LSTM networks. In particular, they presented eDemand, which is an energy consumption forecasting model which employs LSTM networks and an improved sine cosine optimization algorithm for accurate and robust building energy consumption forecasting. Live energy consumption data was obtained from an academic building, the Indian Institute of Technology in Bombay, and it is used to forecast short-term, mid-term, and long-term energy consumption. The conducted experiments revealed that the proposed model outperforms the state-of-the-art energy consumption forecast models according to different evaluation metrics.

Also, a study on Deep Reinforcement Learning (DRL) techniques for building energy consumption forecasts was proposed in [67]. Very little is known about DRL techniques in forecasting building energy consumption. A case study of an office building was presented and three commonly-used DRL techniques to forecast building energy consumption are used: Asynchronous Advantage Actor-Critic (A3C), Deep Deterministic Policy Gradient (DDPG), and Recurrent Deterministic Policy Gradient (RDPG). The objective of the paper was to investigate the potential of DRL techniques in building energy consumption predictions. A comprehensive comparison between the proposed DRL models and common supervised models was provided. Experimental results showed that DDPG outperformed supervised models both in single-step ahead prediction and multi-step ahead prediction. RDPG model did not have advantages over DDPG in single-step ahead prediction, yet led to evident accuracy improvement in multi-step ahead prediction. A3C led to poor performance both in single-step ahead prediction and multi-step ahead prediction, indicating that the technique is not adequate for forecasting building energy consumption.

Producing a model for every single customer is not very scalable (high computational complexity)

and a model that is able to use the information about a customer as features and can generalize over a customer base would be the best solution for an energy company. The next few recent works try to use novel and advanced techniques to try out this approach.

A novel deep ensemble learning-based probabilistic load forecasting framework is proposed in [120] to quantify the load uncertainties of individual customers. The presented framework employed the profiles of different customer groups and integrated them into the understanding of the task. Specifically, customers are clustered into separate groups based on their profiles and a multitask representation learning approach is employed on these groups simultaneously. This technique led to better feature learning across groups and it was particularly useful to improve the performance of predicting residential demand response and managing home energy in smart grids. However, this technique in order to be applicable needs customers' personal information and a large customer base in order to be effective.

[123] proposed a novel day-ahead residential load forecasting method based on feature engineering, pooling, and a hybrid DL model. Feature engineering was performed using two-stage preprocessing on data from each user, i.e., first decomposition and then multi-source input dimension reconstruction. Subsequently, the pooling operation was adopted to merge data from both the target user and its interconnected users, in a descending order based on mutual information. Finally, a hybrid model with two input channels was developed by combining LSTM with a self-attention mechanism. The case studies were conducted on a practical dataset containing multiple residential users. The proposed load forecasting method achieved the best performance with a four-user data pool, 49 time steps, and 24 feature dimensions. The optimal performance corresponded to 15.33%, 56.86 kW, and 82.50 kW in terms of MAPE, MAE, and RMSE, respectively. The proposed method was demonstrated to be an effective choice for day-ahead residential load forecasting.

[80] proposed a daily, weekly, and monthly energy consumption prediction model for single customers by using a TFT. The study used a TFT which considered both primary and valuable data sources, and batch training techniques. A data set of 169 customers have been considered in the study and the TFT was tested on data from only one customer for demonstrating the effectiveness of the customer-based predictive model, which has the advantage that does not depend on overall energy consumption. The model's performance has been related to the LSTM, interpretable LSTM, and TCN models. The overall symmetric MAPE (sMAPE) of LSTM, interpretable LSTM, TCN, and proposed TFT remained at 29.78%, 31.10%, 36.42%, and 26.46%, respectively. The sMAPE of the TFT has proved that the model has performed better than the other DL models. This is a very smart recent work in which by using time series of also different customers is able to improve the quality of forecasts with respect to other models for a single customer.

2.5 Electricity production forecasting

Electricity production forecasting is a classic problem in the time series field. In particular, recently the importance of renewable energy sources is raised due to the clear effects of climate change. Understanding how much energy the systems based on renewable energy sources can produce with respect to their maximum production capacity is crucial. However, this is quite difficult to forecast since depends on future natural phenomena. There are different studies in the literature that distinguishes between different renewable energy sources, such as photovoltaic (PV), wind, geothermal, biomass, and hydropower. In this section, the techniques for PV electricity production forecasting are presented.

PVGIS[5] is a tool of the EU-Joint Research Center that provides information about solar radiation and PV system performance for any location in Europe and Africa, as well as a large part of Asia and America. PVGIS uses high-quality solar radiation data obtained from satellite images, as well as ambient temperature and wind speed from climate reanalysis models. It is a free tool that allows specifying the details of a PV plant to obtain its potential generation. It is useful to get an indication of the potential generation but it is not accurate as the methods presented in the literature. In fact, specific methods can focus on specific plants taking into account historical data from which to extract the specific production performance and consider the most recent weather forecasts available.

[5] `https://joint-research-centre.ec.europa.eu/pvgis-online-tool_en`

[53] reviewed the theory behind the forecasting methodologies, and presented a number of successful applications of solar forecasting methods for both the solar resource and the power output of solar plants at the utility-scale level. Some examples of the presented approaches are Regressive methods, ANNs, Numerical Weather Prediction (NWP), and hybrid methods incorporating two or more techniques.

Zamo et al. in 2014 presented a pair of articles proposing a benchmark of statistical regression methods for short-term forecasting of PV electricity production. The first one treated deterministic forecast of hourly production [121], and the second one probabilistic forecast of daily production [122]. The proposed benchmark designated Random Forest as the best forecast model for hourly PV production with a short lead time (from 28 to 45 h). Their results also suggested that the RMSE can be reduced to about 5.8% by first forecasting the production for each individual power plant and then summing these forecasts up. For probabilistic forecasts of 2 days ahead daily production, quantile regression (QR) based forecasts performed significantly better than the climatology, with a CRPS (continuous ranked probability score) lowered by up to 50%. For most power plants, a QR-based forecast performs better than the others. But the most accurate forecast may vary from one power plant to another and with the number of forecast quantiles.

[9] presented a review of PV power forecasting up to 2016. Forecasting techniques such as regressive methods, ANN, KNN, SVM, Random Forest, and hybrid methods are presented. Most recent papers used ML techniques, due to the ease of modeling without the need of knowing PV plant characteristics, since these approaches are able to learn them from data. Also, spatial and temporal horizons and performance metrics are discussed. The forecast horizon where most research has been done is the day ahead. The reason is that most of the energy is traded in day-ahead markets when planning and unit commitment takes place. They discussed that spatial averaging is very used since it reduces the variability of the solar resource and generates regional forecasts that are more reliable than single-site ones. This is caused by the smoothing effect, which cancels errors with opposite signs in different PV plants.

Barbieri et al. in [12] found out that ANNs and SVM are appropriate approaches for short-term horizons and NWP are better suited for longer horizons. In fact, while a probabilistic method based on historical data may be valuable for very long-term forecasts, such an approach cannot take into consideration the complex variations of the cloud cover causing short-term sunlight disruptions. Only a deterministic atmospheric modeling approach can deal with the stochastic changes of solar radiance during the day. Within this type of model, NWP data-based models are well adapted for day-ahead forecasts but suffer from a too coarse temporal resolution. Sky imagers are a precious tool to identify cloud types and anticipate the impact of shading on PV power generation. They concluded by introducing some future works as the elaboration of algorithms that can calculate cloud cover and classify clouds using online data and a fine sampling period. In addition, measuring precisely the effects of each type of cloud on solar irradiance could greatly help in improving the results.

In [25], Das et al. based on the studies dated up to 2018 found out that ANN and SVM-based forecasting models performed well under rapid and varying environmental conditions. In addition, most of the studies adopted numerous techniques to develop their forecasting model in order to obtain better accuracy. Moreover, a considerable number of studies classified the forecasted day into different categories based on the weather conditions using several techniques and then developed the forecasting model. However, the range of the observed error was remarkably high due to different weather conditions. For performing good predictions and minimizing errors, the separate sub-model for each weather condition has to perform well.

[96] classified solar PV forecasting methods into three major categories, that are time-series statistical, physical, and ensemble methods. Among the most used methods, there are ANN and SVM thanks to their ability in solving complex and non-linear forecasting models. The metrics assessment shows that Artificial Intelligence (AI) models could decrease the error compared to other statistical approaches. The ensemble methods have been introduced recently and thanks to their ability to merge linear and non-linear techniques enhanced the accuracy and performance in comparison with individual models. The standard metrics that are used for evaluating solar prediction accuracy were presented for specific applications, they allow the selection of the appropriate solar forecasting approaches to

ensure better performance.

[26] presented a literature review on big data models for solar PV electricity generation forecasts, aiming to evaluate the most applicable and accurate state-of-the-art techniques to the problem. They included also the motivation behind each project proposal, and the characteristics and quality of data used to address the problem, among other issues. They affirmed that the prediction of solar electricity generation is currently an ongoing academic research question. ML is widely used, and approaches based on neural networks are considered the most accurate. Extreme learning machine (ELM) was also a great addition and it has reduced training time and raised precision.

[1] investigated the accuracy, stability, and computational cost of Random Forest and Extra Trees techniques for predicting the hourly PV generation output. They compared the performance of the proposed methods with SVR. They proved that all developed models have comparable predictive power and are equally applicable for predicting hourly PV output. Despite their comparable predictive power, Extra Trees outperformed Random Forest and SVR in terms of computational cost. They concluded by affirming that the stability and algorithmic efficiency of Extra Trees makes this technique an ideal candidate for wider adoption in PV output forecasting.

In [4], Ahmed et al. reviewed and evaluated contemporary PV solar power forecasting techniques. They noticed through correlation analysis that solar irradiance is the most correlated feature with PV output and consequently weather classification and cloud motion study are crucial operations for optimal results. In addition, they stated that the best data-cleaning processes are normalization and wavelet transforms, and that augmentation using generative adversarial networks is recommended for network training and forecasting. Furthermore, they analyzed also that the optimization of inputs and network parameters can be done by using GAs and particle swarm optimization. They determined that ensembles of ANNs are the best approach for forecasting short-term PV power.

[58] examined the performance of the LSTM method in Turkey's electricity production estimation and determined the optimization technique that provides the best performance in the LSTM estimation method. It was observed that the energy production estimation of LSTM and Adam optimization technique achieved successful results.

In [44], Gellert et al. proposed and evaluated a context-based technique to anticipate electricity production and consumption in buildings. They focused on a household with PVs and an energy storage system. They analyze the efficiency of Markov chains, stride predictors, and also their combination into a hybrid predictor in modeling the evolution of electricity production and consumption. Experimental results showed that the best predictor is the Markov chain configured with an electric power history of 100 values, a context of one electric power value, and an interval size of 1.

A GA-based SVM (GASVM) model for short-term power forecasting of residential-scale PV systems is proposed in [107]. The GASVM model classified the historical weather data using an SVM classifier initially and later it is optimized by the GA using an ensemble technique. Experimental results demonstrated that the proposed GASVM model outperformed the conventional SVM model by a difference of about 669.624W in the RMSE value and 98.7648% of the MAPE error.

In [128], a hybrid model (SDA-GA-ELM) based on ELM, GA, and customized similar day analysis (SDA) has been developed to predict hourly PV power output. In the SDA, the Pearson correlation coefficient is employed to measure the similarity between different days based on five meteorological factors, and the data samples similar to those from the target forecast day are selected as the training set of ELM. In the ELM, the optimal values of the parameters are searched by the GA to improve the prediction accuracy. The results show that the SDA-GA-ELM model has higher accuracy and stability than other tested approaches in day-ahead PV power prediction, such as ELM, SVM, SDA-ELM, and SVM-ELM.

[70] presented case studies on forecasting PV power production and electricity demand in Portugal. They studied an ensemble of different ML methods (SVM, Random Forest, LSTM, and ARIMA) to exploit the growing collection of energy supply and demand records. The ensemble used only electricity data to forecast since only this data is available online for any forecasting horizon. The ensemble method was based on offline training and online forecasting, by applying the most recent power measurements to trained models. The different ML methods performed different non-linear transformations to the same electricity data, thus introducing diversity in the ensemble. To assess

the forecasting performance of the system, they considered two forecasting horizons relevant to the Internal Electricity Market, namely 36 hours ahead, relevant to the single day-ahead coupling, and 2 hours ahead, relevant to the single intraday coupling. The forecasting performance using only electricity data is compared with state-of-the-art models and improves the reference accuracy in their case studies. Since the ensemble relies only on energy data, the results showed that ML methods are useful to exploit energy big data for efficient energy forecasting systems. As demonstrated by other papers, weather information such as solar irradiance has a high correlation with energy production, and when studying a few plants, instead of all the ones in Portugal, this information has a relevant impact on performance.

A novel hybrid method for deterministic PV power forecasting based on wavelet transform (WT) and CNN is proposed in [109]. WT is used to decompose the original signal into different frequency series. CNN is employed to extract the nonlinear features and invariant structures exhibited in each frequency. A probabilistic PV power forecasting model that combines the proposed deterministic method and spine quantile regression is developed to statistically evaluate the probabilistic information in PV power data. Statistical results showed that the average MAPE, RMSE, and MAE of the proposed deterministic model outperformed the compared benchmarks in terms of seasons, forecasting horizons, and PV power locations.

Day-ahead power output time-series forecasting methods are proposed in [43], in which the ideal weather type (sunny day) and non-ideal weather types (rainy days, windy days, and foggy days) have been separately discussed. For ideal weather conditions, a forecasting method is proposed based on meteorology data of the next day using LSTM networks. For non-ideal weather conditions, time-series relevance and specific non-ideal weather type characteristic are considered in the LSTM model by introducing adjacent day time-series and typical weather type information. Specifically, daily total power, which is obtained by the discrete grey model (DGM), is regarded as input variables and applied to correct power output time-series prediction. Prediction performance comparison between proposed methods with traditional algorithms revealed that the RMSE accuracy of forecasting methods based on LSTM networks can reach 4.62% for ideal weather conditions. For non-ideal weather conditions, the dynamic characteristic is effectively described by proposed methods and the proposed methods obtained superior prediction accuracy. This is an interesting study on how to treat the problem in two subcases, however, for real applications, a single network able to adapt to weather conditions by itself is needed.

In [111], a CNN, a LSTM network, and a hybrid model based on CNN and LSTM models were proposed by Wang et al and they are applied to the data of the DKASC (Dattajirao Kadam Arts, Science and Commerce) college PV system. The results showed that when the input sequence is increased, the accuracy of the model is also improved, and the prediction effect of the hybrid model is the best, followed by that of the CNN. While the LSTM network had the worst prediction effect, the training time was the shortest.

In [112], a hybrid DL model (LSTM-CNN) is proposed and applied to PV power prediction. In the proposed hybrid prediction model, the temporal features of the data are extracted first by the LSTM network, and then the spatial features of the data are extracted by the CNN model. The results showed that the hybrid prediction model had a better prediction effect than the single prediction models in isolation, and the proposed hybrid model is also better than the CNN-LSTM (extract the spatial characteristics of data first, and then extract the temporal characteristics of data).

A hybrid DL model combining wavelet packet decomposition (WPD) and LSTM networks is proposed in [63]. The hybrid DL model is utilized for one-hour-ahead PV power forecasting with five-minute intervals. WPD is first used to decompose the original PV power series into sub-series. After the decomposition, four independent LSTM networks are developed for the sub-series. Finally, the results predicted by each LSTM network are reconstructed and a linear weighting method is employed to obtain the final forecasting results. Experimental results show that the proposed hybrid DL model exhibits superior performance in both forecasting accuracy and stability with respect to LSTM, RNN, GRU, and MLP.

In [75], different kinds of neural networks for short-term output PV power forecasting have been developed and compared: LSTM, Bidirectional-LSTM (BiLSTM), GRU, Bidirectional-GRU (BiGRU),

One-Dimension CNN (CNN1D), as well as other hybrid configurations such as CNN1D-LSTM and CNN1D-GRU. A database of the PV power produced by the microgrid installed at the University of Trieste (Italy) is used to train and comparatively test the neural networks. The performance has been evaluated over four different time horizons, for one-step and multi-step ahead. The results show that the investigated neural networks provide very good accuracy, particularly in the case of a 1-minute time horizon with one-step ahead (correlation coefficient is close to 1), while for the case of multi-step ahead (up to 8 steps ahead) the results are found to be acceptable (correlation coefficient ranges between 96.9% and 98%). The new advanced neural network algorithms are able to lead to acceptable accuracy in the case of cloudy days. However, even though the models have great potential for fine time granularity, these time ranges are quite limited and they should be proved to be effective also on longer ranges to be used by PV plant owners.

[71] aimed to predict hourly day-ahead PV power generation by applying the TFT model. TFT incorporates an interpretable explanation of temporal dynamics and high-performance forecasting over multiple horizons. The proposed forecasting model has been trained and tested using data from six different facilities located in Germany and Australia. The results have been compared with other algorithms like ARIMA, LSTM, MLP, and XGBoost. The use of TFT has been shown to be more accurate than the rest of the algorithms in forecasting PV generation in the tested different facilities. The importance of the decoder and encoder variables has been also calculated, revealing that solar horizontal irradiation and the zenith angle are the key variables for the model. This model was proved to be effective also in this task, showing the superiority of the TFTs over standard DL models, even though is quite more complex and requires a larger amount of data to achieve good prediction results.

3 System Model

In this chapter, the model of the proposed system is presented. In the first section, an in-depth presentation of the designed system architecture is provided. The different components of the system with their functionalities and interactions are described. Lastly, the modules focused on the three use cases of interest are treated more in detail in dedicated sections. After this chapter, it will be clear what the main parts of the system are and how they cooperate to accomplish the various use cases of interest.

3.1 System architecture

The use case diagram for the proposed system is presented in figure 3.1. Users can interact with the system via dedicated APIs. To be authorized, a user must be authenticated and have an account with an active subscription. Users can request the following asynchronous operations:

- Load new data with a create or update logic specifying the type of data;
- Train new models based on available data for a specific use case specifying the time granularity (hourly or daily). The supported specific use cases for the models that will be used in the dedicated forecast requests are electricity demand forecasting, consumption baseline forecasting, and electricity production forecasting;
- Forecast future data for a specific use case using a specified model for a certain time granularity, for a certain starting date and time horizon.

In addition, a task scheduler periodically triggers the performance evaluation of the available models and if needed triggers the models re-training for having up-to-date models with respect to the user data so that they might perform better on future forecasts.

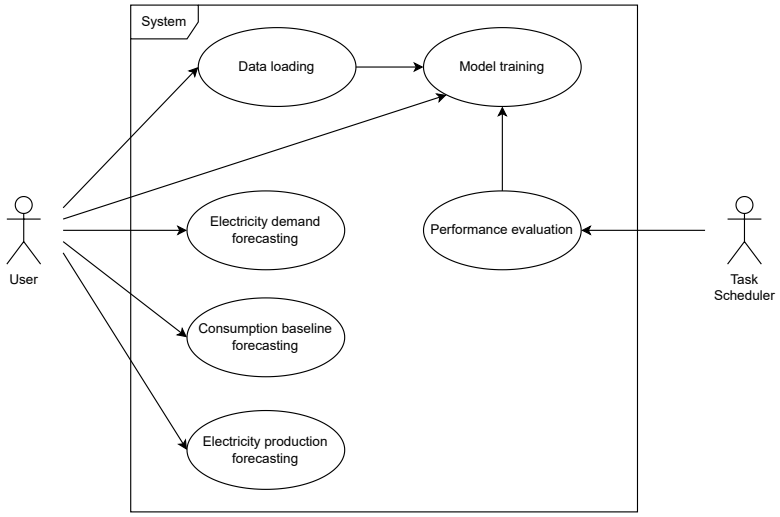


Figure 3.1: The use case diagram for the proposed system.

For achieving the presented use cases the system architecture shown in figure 3.2 was designed. It is composed of the following components:

- Authentication layer: manages the authentication process and verifies user permissions;

- API layer: manages the user interactions with the system;
- Task manager: coordinates the execution of the tasks;
- Data loading task: parses and stores the users' loaded data;
- ML models interface: handles the interactions with the ML models storage where the trained ML models are stored;
- Model training task: trains new models based on available data;
- Forecasting task: forecasts future data using a specified model;
- Task scheduler: manages the periodic scheduled tasks;
- Performance evaluation task: evaluates the performance of the available models and if needed triggers the models re-training.

The system interacts with the following external components:

- Users data storage: used to store and retrieve user data, including account data;
- Data storage: used to store and retrieve the users' loaded data;
- Weather data storage: used to retrieve the weather data;
- ML models storage: used to store and retrieve the trained models;
- Forecasts storage: used to store and retrieve the computed forecasts.

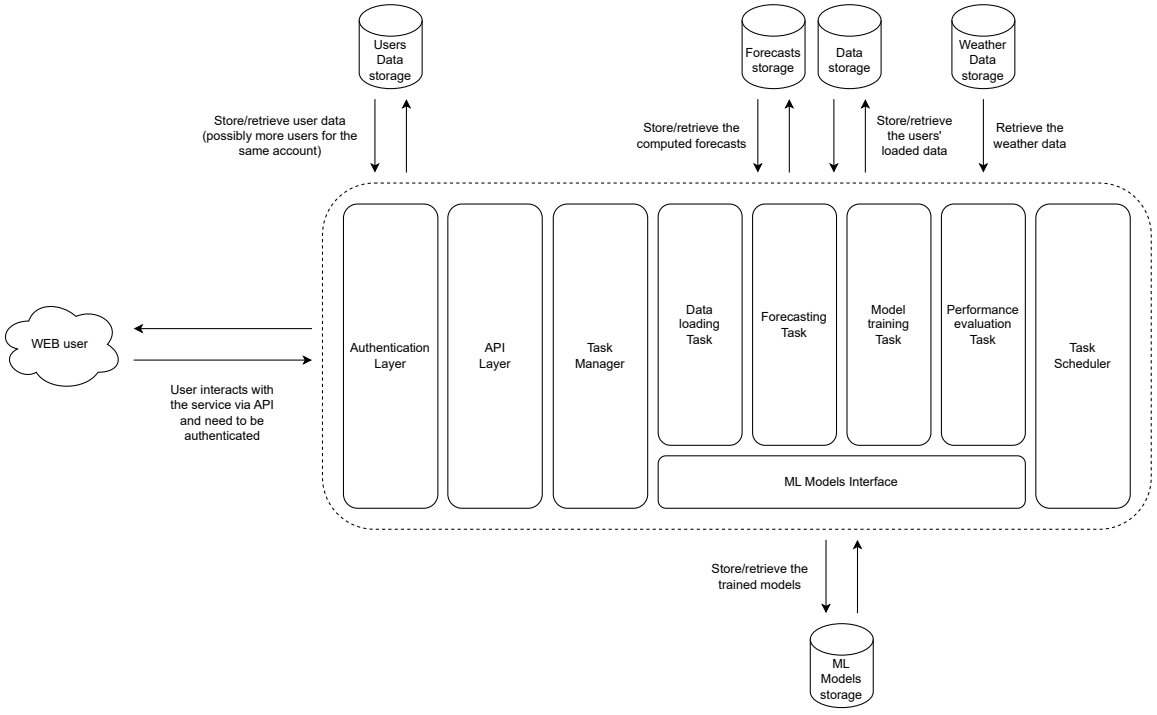


Figure 3.2: The architecture designed for the proposed system.

Figure 3.3 shows the interactions among the components of the designed architecture for achieving the presented use cases. In the following subsections, the logic for each use case is explained with the relative components' interactions.

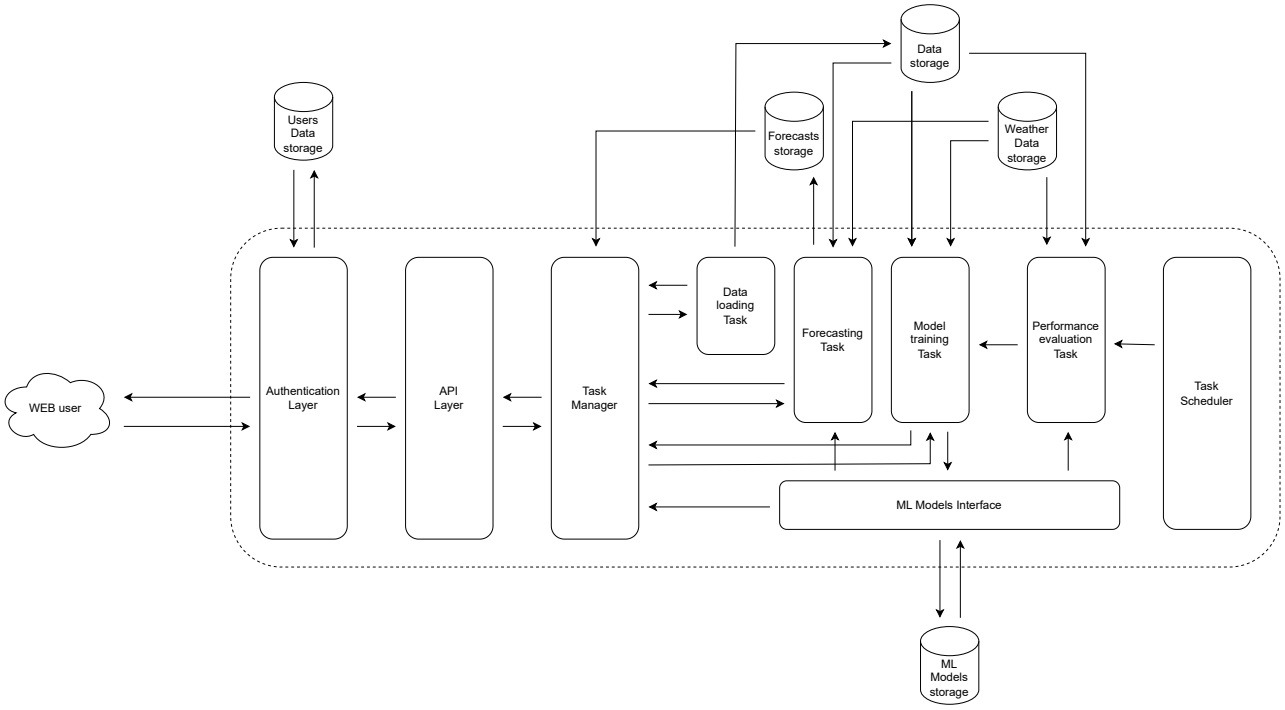


Figure 3.3: The interactions among the components in the designed architecture.

3.1.1 Data loading

The diagram representing the components' interactions for achieving the data loading use case is presented in figure 3.4.

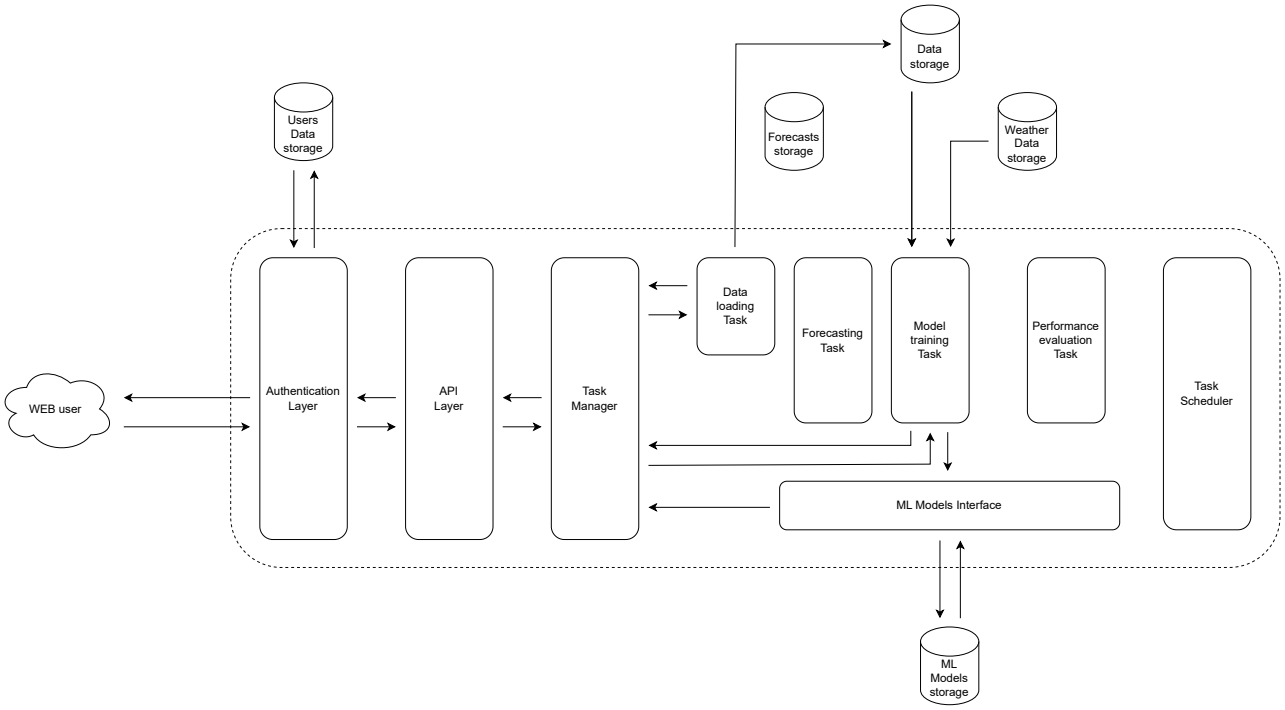


Figure 3.4: The interactions among the components for achieving the data loading use case.

The user sends the data to the system, for example uploading CSV files. First, the authentication layer verifies the authentication of the user and checks whether it has a subscription to load the data. The endpoint dedicated to data loading takes charge of the user request and provides the task manager with the details for creating a data loading task. The identifier of the task is then returned to the

user, who can request the status of the data loading request and check the result of the operation when completed.

The data loading task starts when the task manager schedules it. As shown by the diagram in figure 3.5, the logic inside this task can be formulated in 2 steps: parse the data and store the data. The data parser block transforms the user data in a system-compatible format and passes it to the data storage block which stores the data inside the data storage. If there is no model for the specific use case for which data is loaded, the task manager will also create a model training task to train all the available models on this data. This allows the system to be ready for the forecast requests on the loaded data. The model training task is described more in detail in subsection 3.1.2.

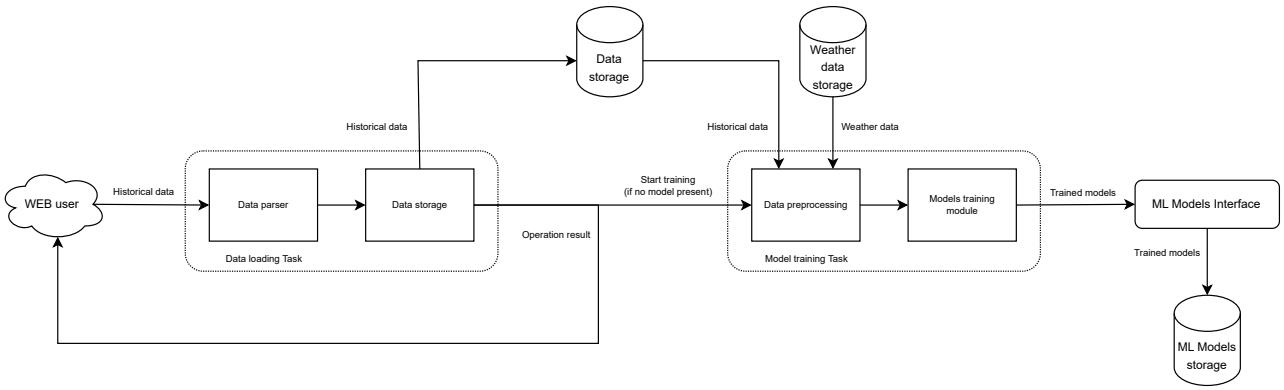


Figure 3.5: Diagram representing the data loading flow.

The sequence diagram representing the complete data loading procedure is presented in figure 3.6.

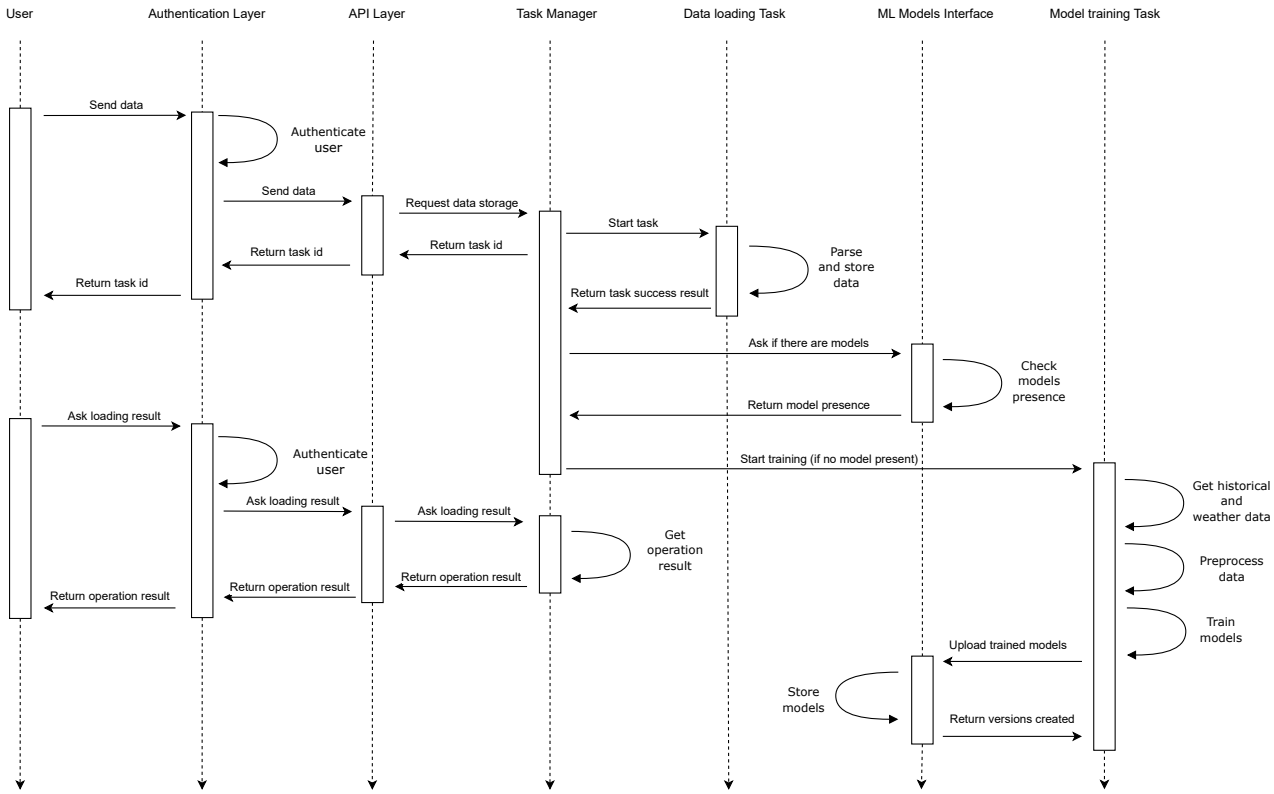


Figure 3.6: Sequence diagram of the data loading procedure.

3.1.2 Model training

The diagram representing the components' interactions for achieving the model training use case is presented in figure 3.7.

The user sends the specification of the model to train to the system. First, the authentication layer verifies the authentication of the user and checks whether it has a subscription to request the training of a model. The endpoint dedicated to model training takes charge of the user request and provides the task manager with the details for creating a model training task. The identifier of the task is then returned to the user, who can request the status of the model training request and get the trained model when completed.

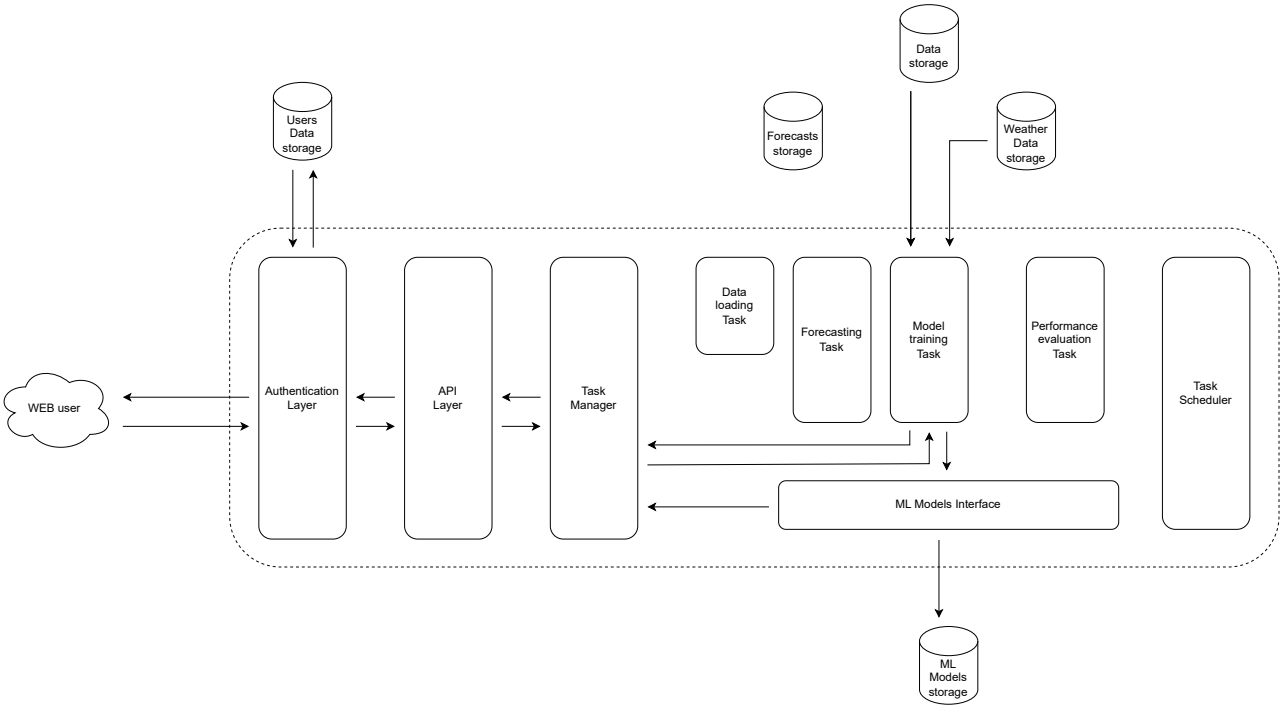


Figure 3.7: The interactions among the components for achieving the model training use case.

The model training task starts when the task manager schedules it. As shown by the diagram in figure 3.8, the logic inside this task can be formulated in 2 steps: preprocess the available data and train the model. The data preprocessing block collects the available data for the specific use case for which the model will be trained, aggregates it correctly, cleans it, and enriches it with weather data. The models training module takes the preprocessed data and trains the model with the provided specifications. The resulting model is then passed to the ML models interface which stores it inside the ML models storage.

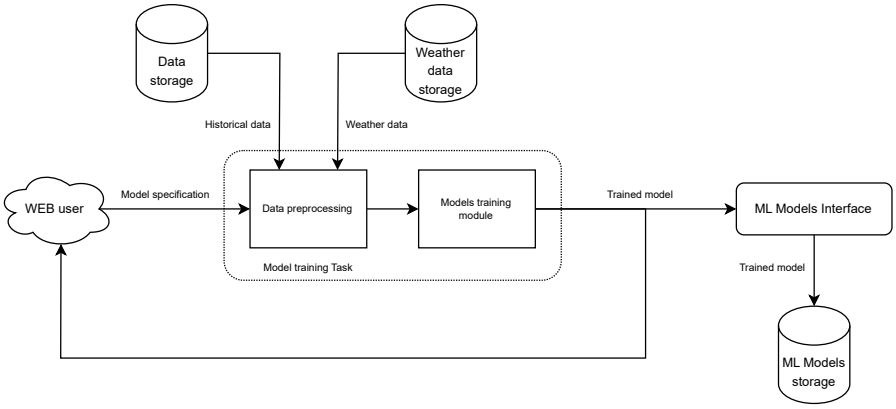


Figure 3.8: Diagram representing the model training flow.

The sequence diagram representing the complete model training procedure is presented in figure 3.9.

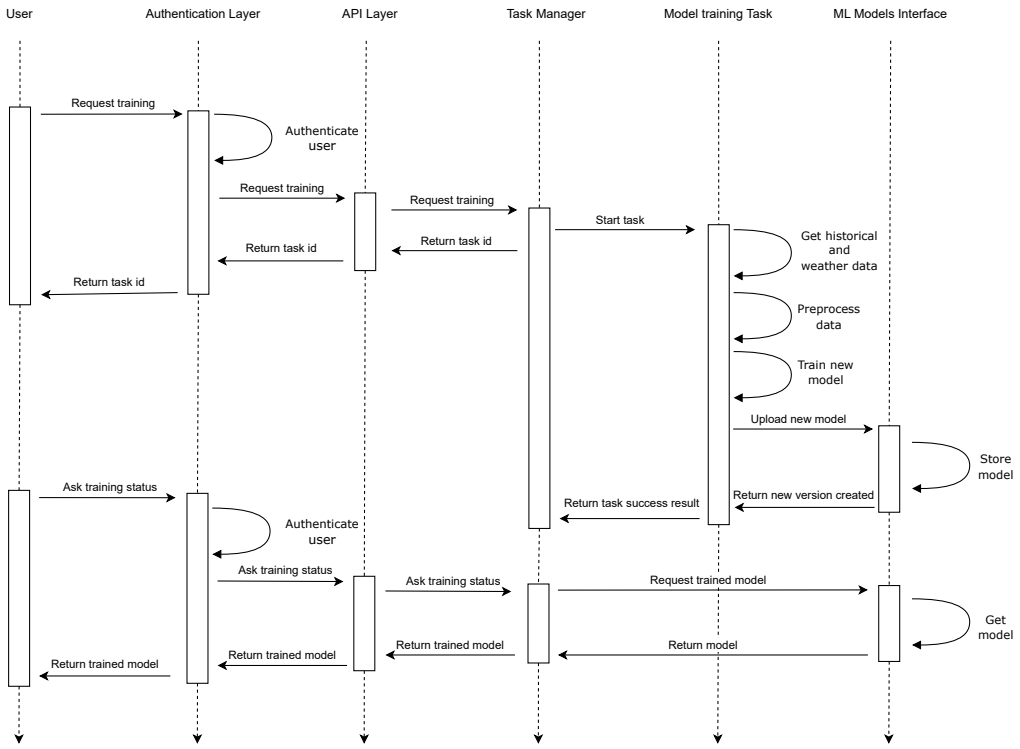


Figure 3.9: Sequence diagram representing the model training procedure.

3.1.3 Forecasting

The diagram representing the components' interactions for achieving the forecasting for a specific use case is presented in figure 3.10.

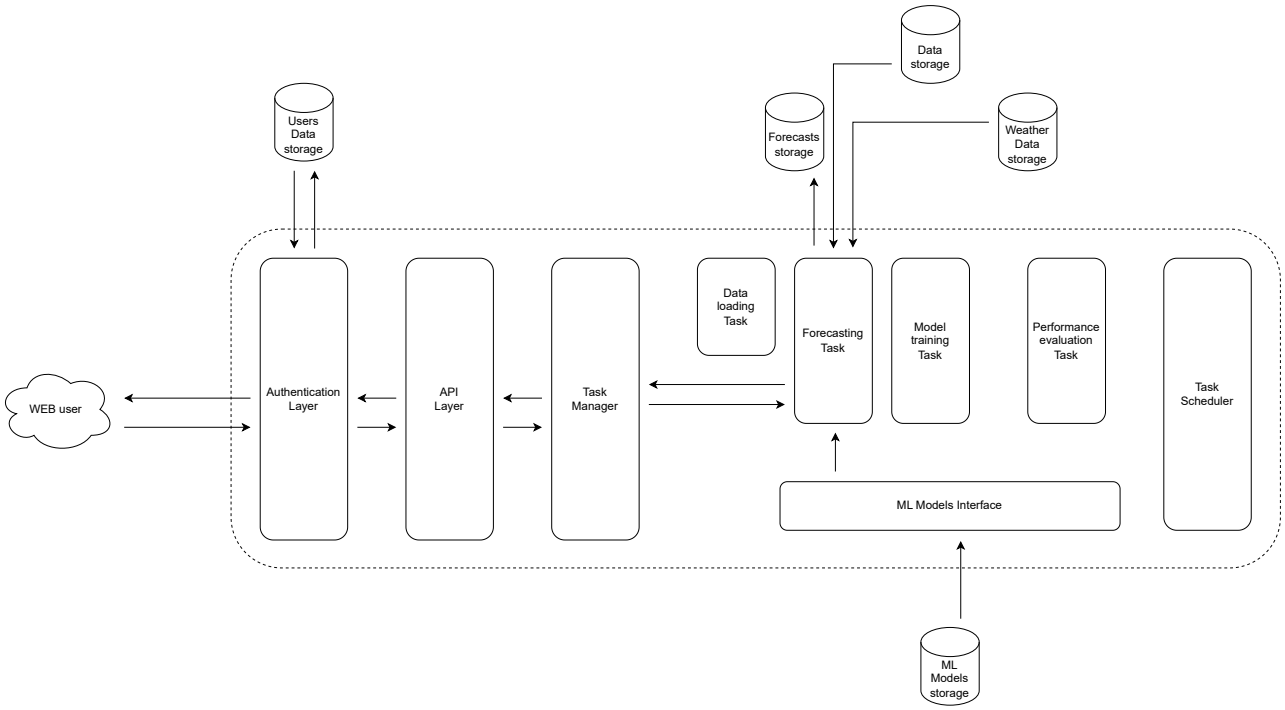


Figure 3.10: The interactions among the components for achieving the forecasting for a specific use case.

The user sends the specification for computing forecasts for a specific use case to the system. First, the authentication layer verifies the authentication of the user and checks whether it has a subscription

to request the forecasts for a specific use case. The endpoint dedicated to forecasting takes charge of the user request and provides the task manager with the details for creating a forecasting task. The identifier of the task is then returned to the user, who can request the status of the forecasting request and get the computed forecasts when completed.

The forecasting task starts when the task manager schedules it. As shown by the diagram in figure 3.11, the logic inside this task can be formulated in 2 steps: preprocess the needed data by the model and forecast future data. The data preprocessing block collects the needed data by the model for the specific use case for which the forecasts will be made, aggregates it correctly, cleans it, and enriches it with weather data. In addition to past weather data also weather forecasts are collected to help produce more accurate future data. The prediction module takes the preprocessed data, requests the specified model, and computes the forecasts. The computed forecasts are then stored in the forecasts storage.

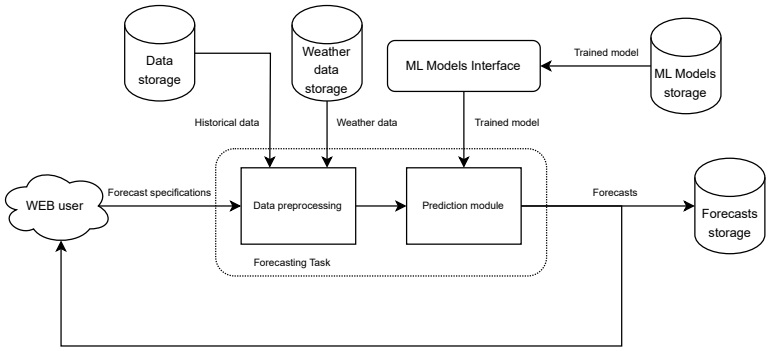


Figure 3.11: Diagram representing the forecasting flow.

The sequence diagram representing the complete forecasting procedure for a specific use case is presented in figure 3.12.

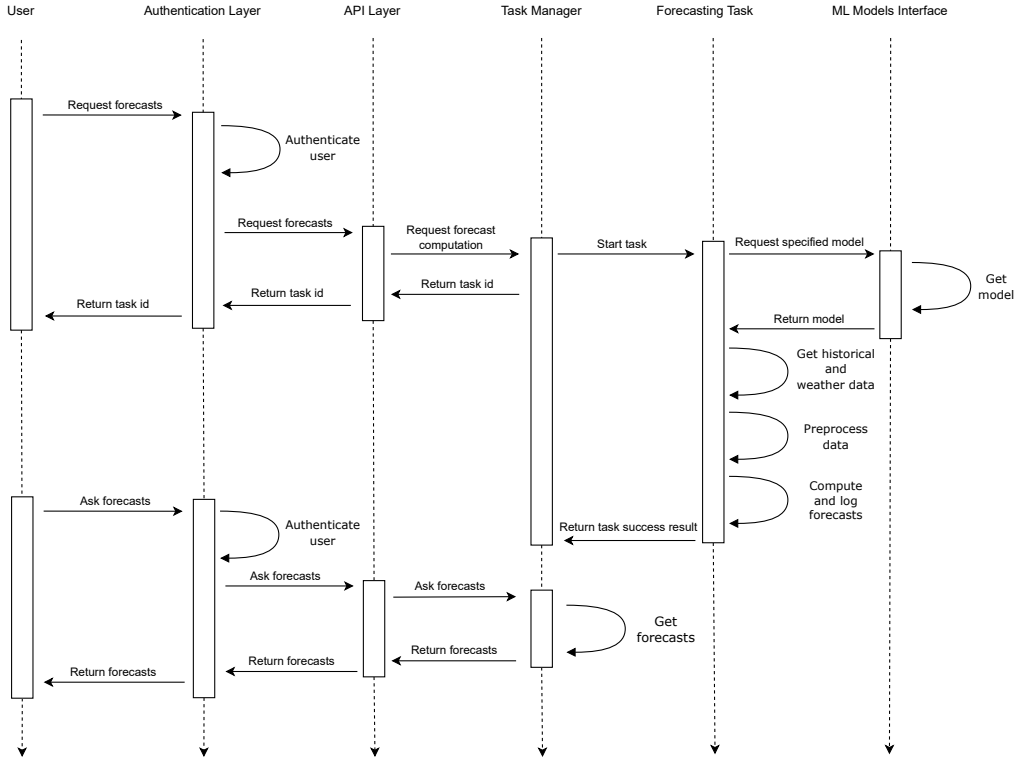


Figure 3.12: Sequence diagram representing the forecasting procedure.

3.1.4 Performance evaluation

The diagram representing the components' interactions for achieving the performance evaluation use case is presented in figure 3.13.

The task scheduler periodically creates and triggers a performance evaluation task. As shown by the diagram in figure 3.14, the logic inside the performance evaluation task can be formulated in 2 steps: preprocess the data and evaluate the performance. The data preprocessing block collects the data for the specific use case for which the model will be trained, aggregates it correctly, cleans it, and enriches it with weather data. The performance evaluation module takes the preprocessed data and evaluates the performance of the last available models. The performance evaluation procedure with the obtained results is described in chapter 5. When performance drops significantly, it triggers the models re-training by creating a model training task for having up-to-date models with respect to the user data so that they might perform better on future forecasts. The model training task is described more in detail in subsection 3.1.2.

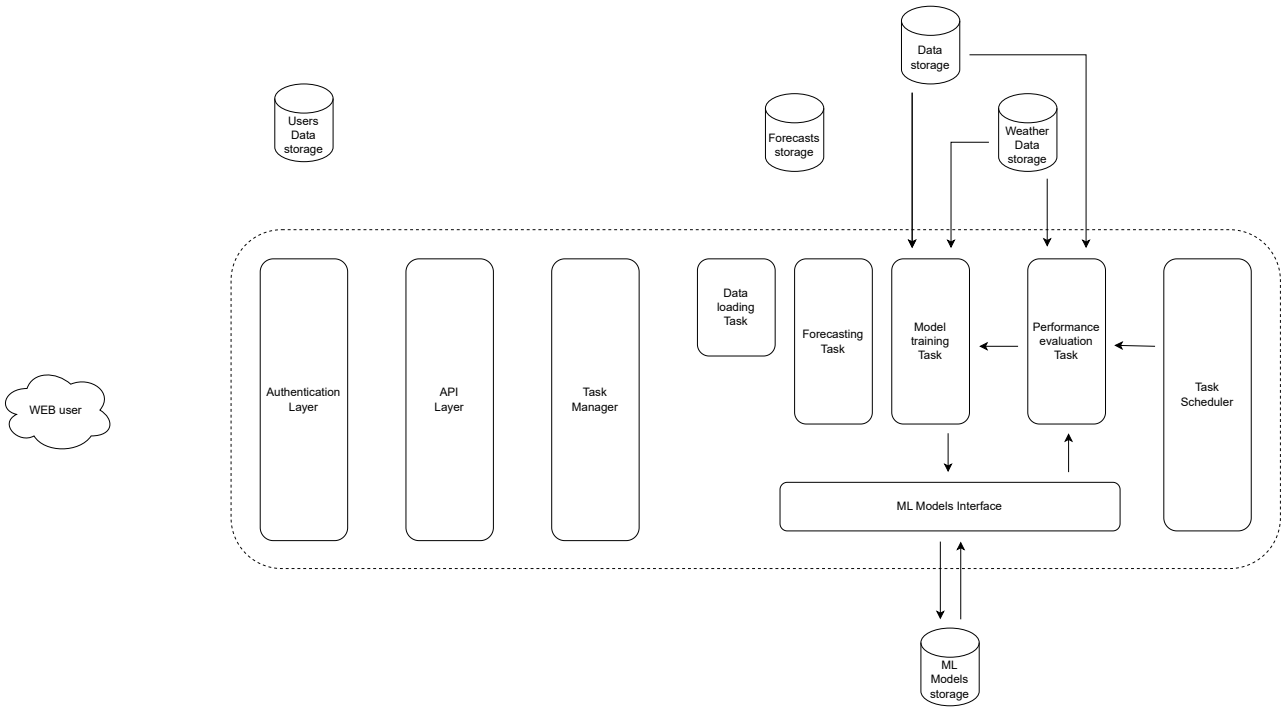


Figure 3.13: The interactions among the components for achieving the performance evaluation use case.

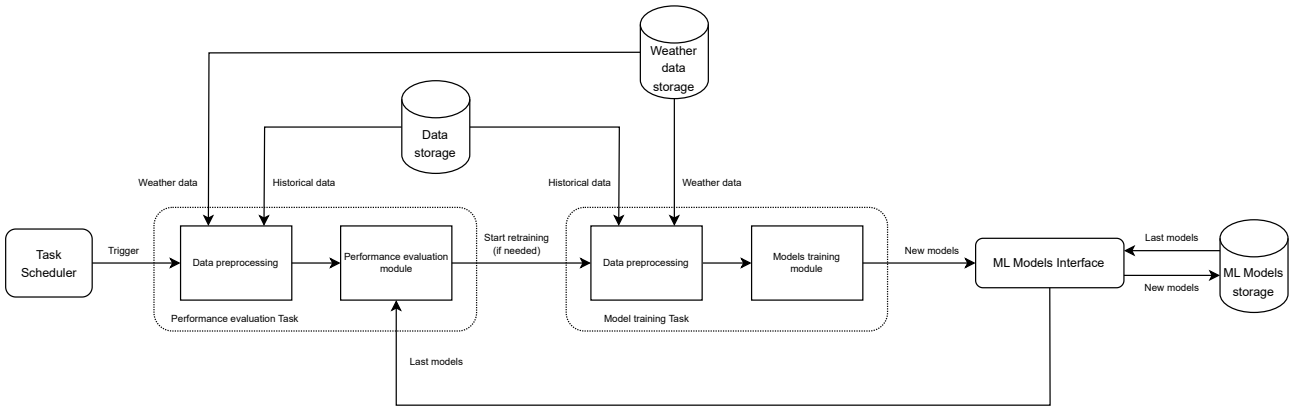


Figure 3.14: Diagram representing the performance evaluation flow.

The sequence diagram representing the complete performance evaluation procedure is presented in figure 3.15.

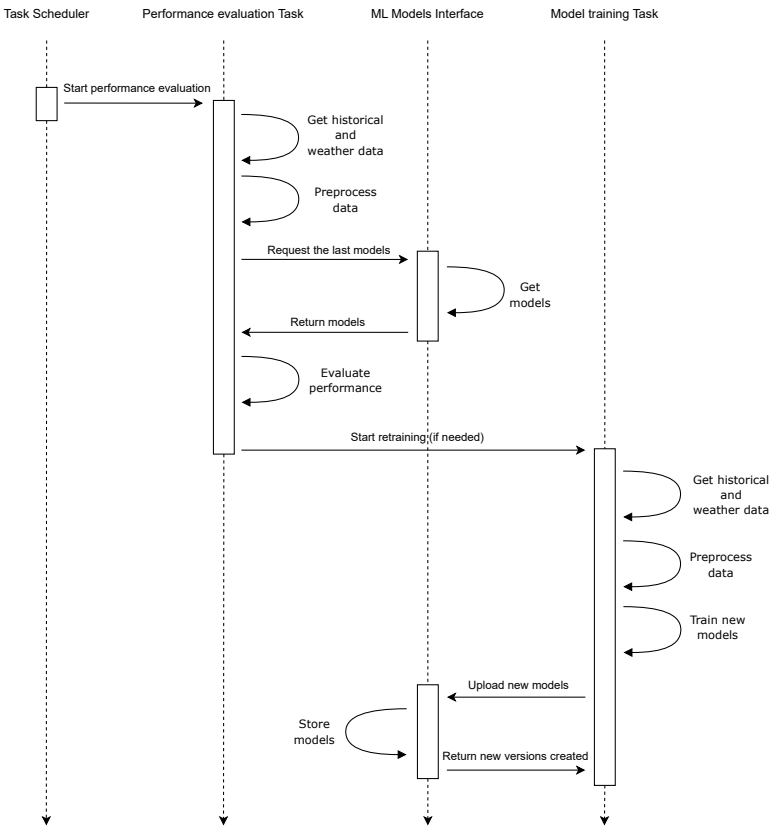


Figure 3.15: Sequence diagram representing the performance evaluation procedure.

3.2 System's common components

In this section, the system's common components across the various specific use cases are described in detail.

The authentication layer manages the user authentication process. In particular, for accessing the system the user needs to be authenticated. When users interact with the system via dedicated APIs, to be authorized they must provide one valid token and have an account with an active subscription. It is also possible to have more users associated with the same account.

The API layer manages the user interactions with the system. Many endpoints are available for managing the users' possible operations, such as data loading, model training, and forecasting requests. Data loading management endpoints consist of requesting the loading of new data and checking the status of the data loading operations. Model training management endpoints consist of requesting the training of new models and checking the status of the model training operations. Forecasting requests management endpoints consist of requesting the forecasts of future data and checking the status of the forecasting requests operations.

The task manager coordinates the execution of the different tasks requested by the users. Where the possible tasks, as described in the previous section, are the following:

- Data loading task, which parses and stores the users' loaded data on the basis of the type of data;
- Model training task, which trains new models based on available data. The specialization of this task for the specific use cases is described in the dedicated sections;
- Forecasting task, which forecasts future data using a specified model. The specialization of this task for the specific use cases is described in the dedicated sections.

For the data loading task, the main blocks are:

- Data parser, which transforms the input user data in a system-compatible format, the expected information in data depends on the specific use case for which data is loaded;

- Data storage, which stores the parsed data inside the data storage with a create or update logic specifying the type of data.

The base blocks across the various specific use cases that are then described more in detail in the following sections are the following:

- Data preprocessing, which retrieves the needed data from the data storage, aggregates it correctly, cleans it by filling the gaps by using a linear interpolation, and enriches it using the weather information retrieved from the weather data storage. For future data forecasts in addition to past weather data also weather forecasts are collected;

- Models training module, which takes the preprocessed data and trains the specified model with the provided specifications;

- Prediction module, which takes the preprocessed data, requests the specified model, and computes the forecasts.

The ML model interface handles the interactions with the ML models storage where the trained ML models are stored. In particular, this component provides the functionalities to store and retrieve models in the ML models storage, This component allows the system to support lots of models for each account and new ones can be added over time. A versioning mechanism is also handled in the model storage. This component allows the system to provide users with the latest available models and choose the best-performing ones.

The task scheduler manages the periodic scheduled tasks. In particular, as described in the previous section, it is present just the performance evaluation task, which evaluates the performance of the available models and if needed triggers the models re-training. The main block of this task is the performance evaluation module, which takes the preprocessed data and evaluates the performance of the last available models using specific use case error metrics such as mean absolute percentage error (MAPE) and mean absolute error (MAE).

3.3 Electricity demand forecasting module

In this section, the electricity demand forecasting module is described. The module handles the specialization of data models, training, and forecasts for the electricity demand forecasting use case.

The UML (Unified Modeling Language) data class for electricity demand forecasting is represented in figure 3.16. The considered attributes are the following: the date and time of the detection, the active incoming to the customers in the past hour, and the number of customers. The basic data is enhanced with the following weather information, the reported value is the average of the value in the last hour: air temperature, apparent temperature, and relative humidity. The mean active incoming per customer is calculated as the division of the total active incoming by the number of customers.

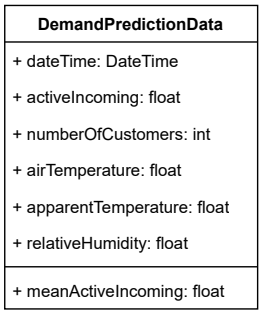


Figure 3.16: The UML data class for electricity demand forecasting.

The schematic representation of the model training procedure for electricity demand forecasting is presented in figure 3.17. It consists of 2 modules: the data preprocessing module and the models training module. The data preprocessing module takes historical aggregated consumption data and historical weather data as input and prepares the data for the model training. The models training module based on the prepared data trains the set of available models for electricity demand forecasting: SARIMA, support vector regressor, hist gradient boosting regressor, extreme gradient boosting regressor, prophet, LSTM, GRU, CNN, TFT, and the combination of different techniques which is also tested. Some baseline approaches which consider repeating past days and weeks, and an AutoML approach are also considered as comparisons.

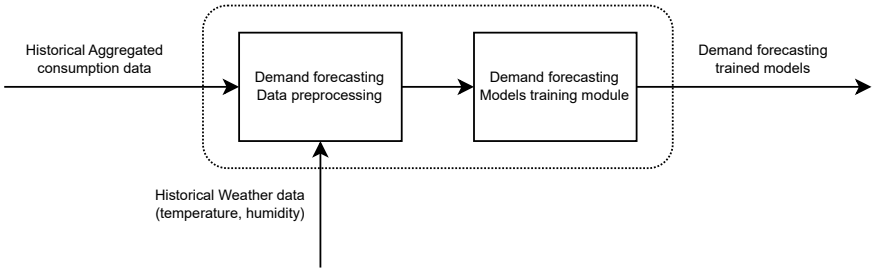


Figure 3.17: The schematic representation of the model training procedure for electricity demand forecasting.

The schematic representation of the forecasting procedure for electricity demand forecasting is presented in figure 3.18. It consists of 2 modules: the data preprocessing module and the prediction module. The data preprocessing module takes forecast specifications, historical aggregated consumption data, historical weather data, and weather forecasts as input and prepares the data for the forecasting. The prediction module based on the prepared data computes the aggregated demand forecasts for the trained available models.

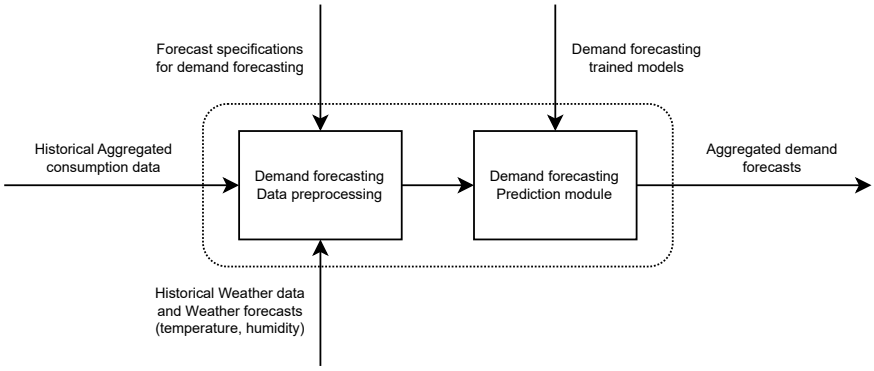


Figure 3.18: The schematic representation of the forecasting procedure for electricity demand forecasting.

3.4 Consumption baseline forecasting module

In this section, the consumption baseline forecasting module is described. The module handles the specialization of data models, training, and forecasts for the consumption baseline forecasting use case.

The UML data class for consumption baseline forecasting is represented in figure 3.19. The considered attributes are the following: the date and time of the detection, the CUPS code (Universal Supply Point Code) which identifies the customer, and the active incoming to the customer in the past hour. In the case of other customer information, this might be used to try to improve the performance of the models. The basic data is enhanced with the following weather information, the reported value is the average of the value in the last hour: air temperature, apparent temperature, and relative humidity.

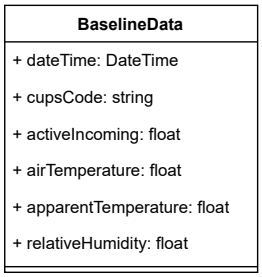


Figure 3.19: The UML data class for consumption baseline forecasting.

The schematic representation of the model training procedure for consumption baseline forecasting is presented in figure 3.20. It consists of 2 modules: the data preprocessing module and the models training module. The data preprocessing module takes historical single customer consumption data and historical weather data as input and prepares the data for the model training. The models training module based on the prepared data trains the set of available models for consumption baseline forecasting: SARIMA, support vector regressor, hist gradient boosting regressor, extreme gradient boosting regressor, prophet, LSTM, GRU, CNN, and TFT. Some baseline approaches which consider repeating past days and weeks, and an AutoML approach are also considered as comparisons.

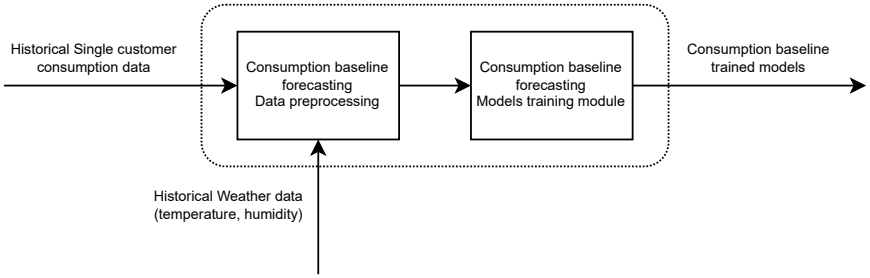


Figure 3.20: The schematic representation of the model training procedure for consumption baseline forecasting.

The schematic representation of the forecasting procedure for consumption baseline forecasting is presented in figure 3.21. It consists of 2 modules: the data preprocessing module and the prediction module. The data preprocessing module takes forecast specifications, historical single customer consumption data, historical weather data, and weather forecasts as input and prepares the data for the forecasting. The prediction module based on the prepared data computes the single customer consumption baseline forecasts for the trained available models.

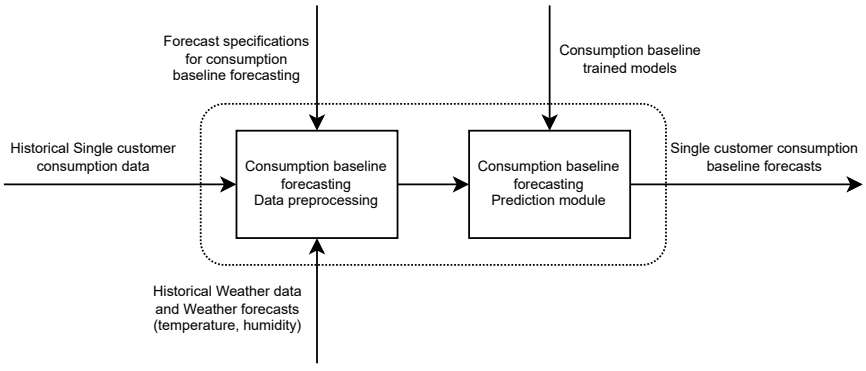


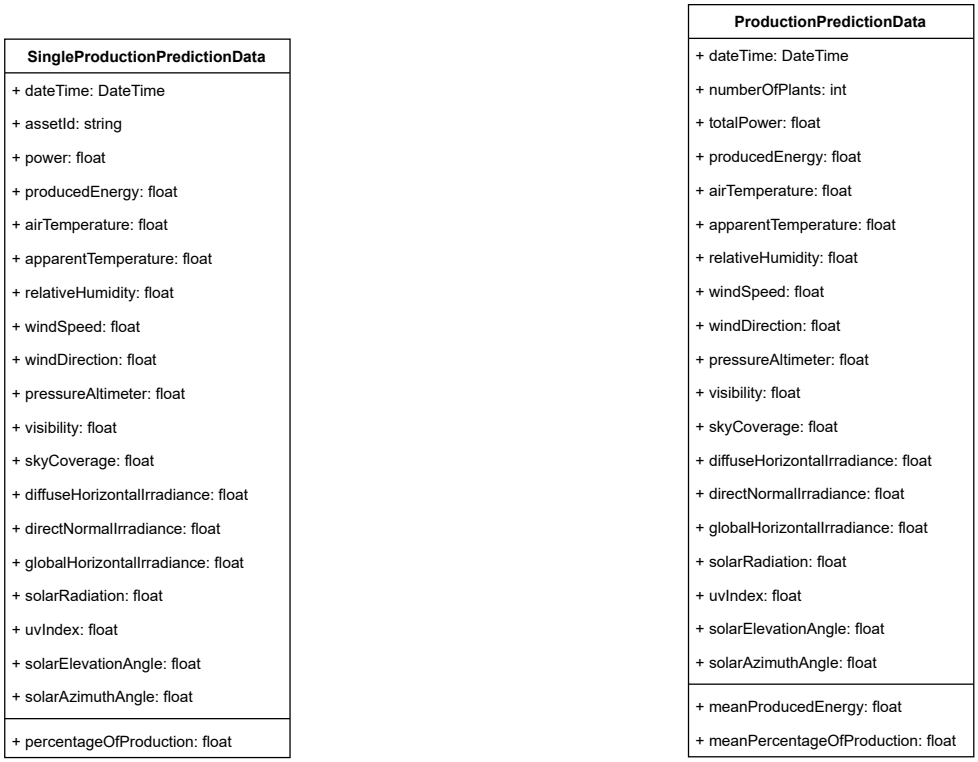
Figure 3.21: The schematic representation of the forecasting procedure for consumption baseline forecasting.

3.5 Electricity production forecasting module

In this section, the electricity production forecasting module is described. The module handles the specialization of data models, training, and forecasts for the electricity production forecasting use case.

The UML data class for a single PV plant data for electricity production forecasting is represented in figure 3.22a. The considered attributes are the following: the date and time of the detection, the id of the asset (intended as the PV plant), the power of the PV plant (intended as its maximum theoretical production power per hour), and the produced energy in the past hour by the PV plant. The basic data is enhanced with the following weather information, the reported value is the average of the value in the last hour: air temperature, apparent temperature, relative humidity, wind speed, wind direction, pressure altimeter, visibility, sky coverage, diffuse horizontal irradiance, direct normal irradiance, global horizontal irradiance, solar radiation, UV index, solar elevation angle, and solar azimuth angle. The percentage of production is calculated by the division of the produced energy by the power of the PV plant.

While the UML data class for the aggregated data over PV plants for electricity production forecasting is represented in figure 3.22b. The considered attributes are the following: the date and time of the detection, the number of PV plants, the total power of the PV plants (intended as the sum of the maximum theoretical production power per hour of all PV plants), and the produced energy in the past hour by the PV plants. The basic data is enhanced with the following weather information, the reported value is the average of the value in the last hour: air temperature, apparent temperature, relative humidity, wind speed, wind direction, pressure altimeter, visibility, sky coverage, diffuse horizontal irradiance, direct normal irradiance, global horizontal irradiance, solar radiation, UV index, solar elevation angle, and solar azimuth angle. The mean produced energy for a single PV plant is calculated as the division of the total produced energy by the number of PV plants. The mean percentage of production is calculated as the division of the total produced energy by the total power of the PV plants.



(a) (b)

Figure 3.22: (a) The UML data class for a single PV plant data and (b) the UML data class for the aggregated data over PV plants for electricity production forecasting.

The schematic representation of the model training procedure for electricity production forecasting

is presented in figure 3.23. It consists of 2 modules: the data preprocessing module and the models training module. The data preprocessing module takes historical PV plants production data and historical weather data as input and prepares the data for the model training. The models training module based on the prepared data trains the set of available models for electricity production forecasting: ARIMA, support vector regressor, hist gradient boosting regressor, extreme gradient boosting regressor, prophet, LSTM, GRU, CNN, TFT, and the combination of different techniques which is also tested. A baseline approach which considers repeating past days and an AutoML approach are also considered as comparisons.

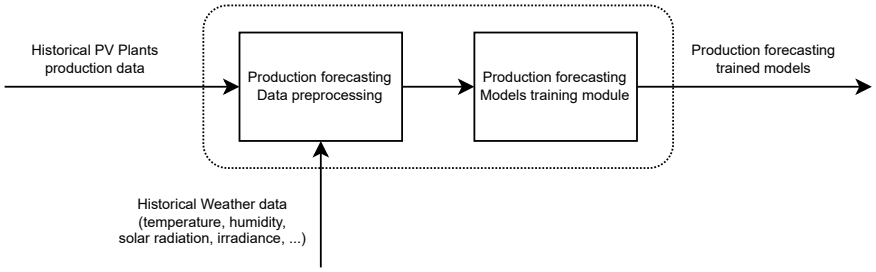


Figure 3.23: The schematic representation of the model training procedure for electricity production forecasting.

The schematic representation of the model training procedure for electricity production forecasting is presented in figure 3.24. It consists of 2 modules: the data preprocessing module and the prediction module. The data preprocessing module takes forecast specifications, historical PV plants production data, historical weather data, and weather forecasts as input and prepares the data for the forecasting. The prediction module based on the prepared data computes the aggregated PV plants production forecasts for the trained available models.

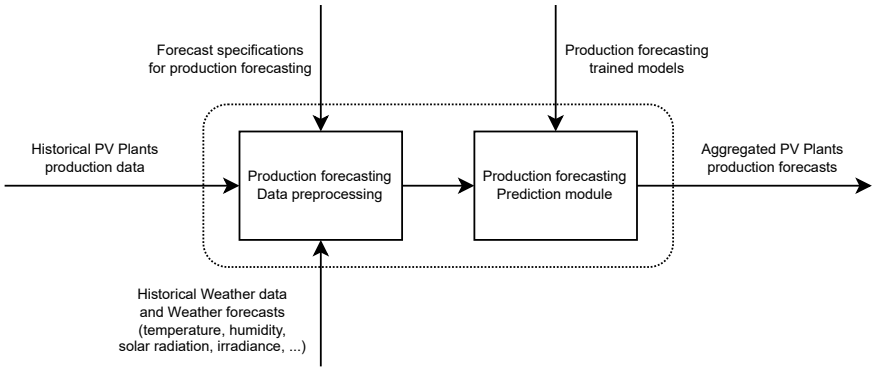


Figure 3.24: The schematic representation of the model training procedure for electricity production forecasting.

4 Prototype Implementation

This chapter presents which components of the system have been implemented and how. Only a subset of the components of the designed architecture in chapter 3 have been implemented. This is because a prototype was developed as a Proof of Concept (PoC) with a focus on key components for validating the core system functionalities for each specific use case. These are the training of models, the forecast of new data, and the evaluation of the performance of the developed models. The remaining components were not implemented since they were not crucial for having a working system, in fact, the overall architecture was designed in order to build a Software as a Service (SaaS) on the implemented core system functionalities.

The first section describes the implementation of the system's common components across the various specific use cases and the others explain the implementation details of the use case-specific modules. After this chapter, it will be clear how the system prototype was implemented and ready for validation and testing phases, which are discussed in chapter 5.

4.1 System's common components

The effectively implemented components of the architecture designed for the proposed system are reported in figure 4.1. The interactions among them are reported in figure 4.2.

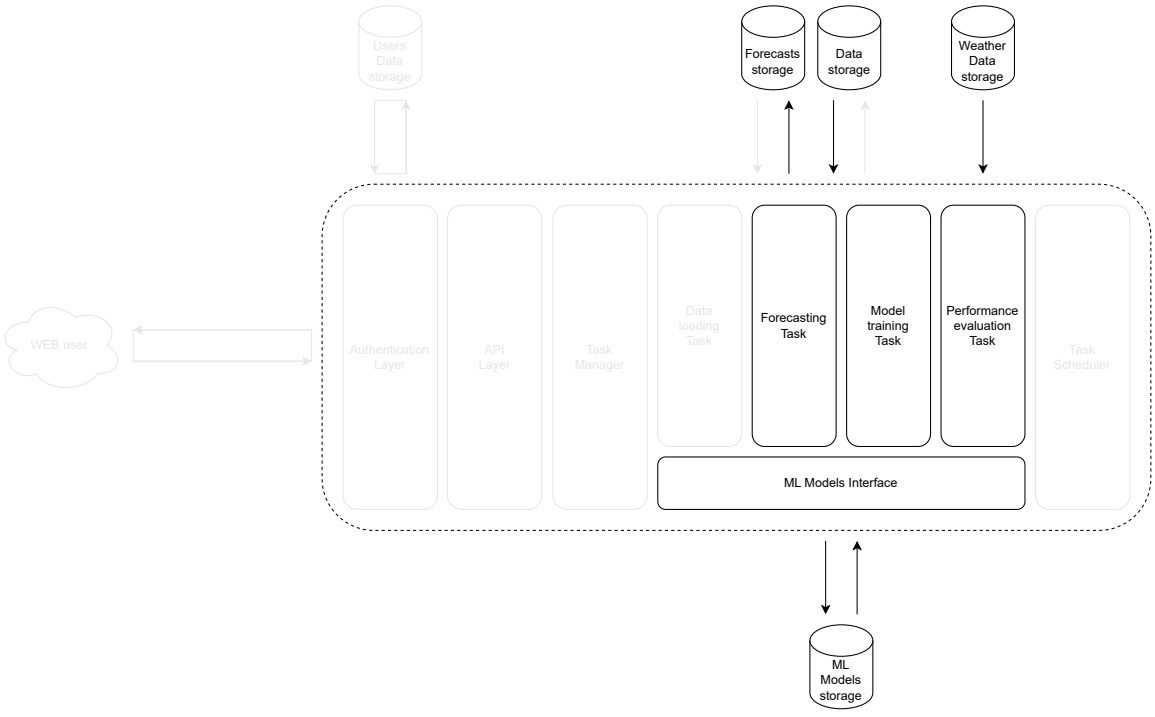


Figure 4.1: The effectively implemented components of the architecture designed for the proposed system.

The ML model interface was implemented using the pickle library[6] for interfacing with the ML models storage for storing the ML models and retrieving the stored ones making them available for prediction and evaluation. The use of the MLflow client[7] was thought of as a possible interface in the

[6] `https://docs.python.org/3/library/pickle.html`
[7] `https://mlflow.org/`

complete system integration for interfacing with the ML models storage.

The model training task trains new models based on available data for the specific use case. The details about the models and their training process for each specific use case are reported in the dedicated sections.

The forecasting task forecasts future data using the available models for the specific use case. The details about the models and their forecasting process for each specific use case are reported in the dedicated sections.

The performance evaluation task evaluates the performance of the available models for the specific use case. The details about the performance evaluation process for each specific use case are reported in chapter 5.

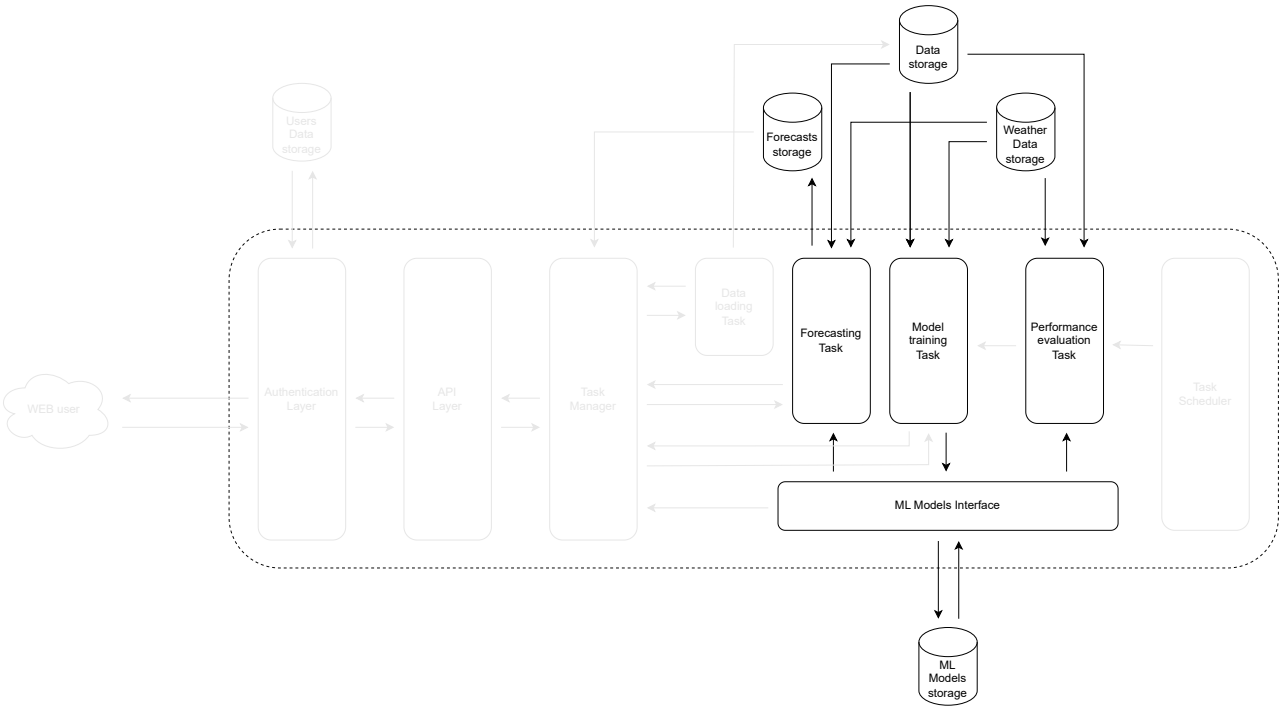


Figure 4.2: The interactions among the effectively implemented components of the architecture designed for the proposed system.

The system interacts with the following external components: data storage, weather data storage, ML models storage, and forecasts storage.

The data storage consists of CSV files containing the data for the different use cases, which are loaded using the pandas library[8]. InfluxDB[9] was thought of as a possible database in the complete system integration for managing the data.

The weather data storage consists of a CSV file and a JSON file containing the weather data, which are loaded using the pandas library. Weather data in the CSV file is manually obtained from Iowa Environmental Mesonet[10] for the weather station in the Murcia airport, and it was used for customers demand forecasting. Weather data in the JSON file is manually obtained from the Weatherbit[11] APIs for a weather station near Murcia airport, these were used for PV plants production forecasting since it provides solar energy data. InfluxDB was thought of as a possible database in the complete system integration for managing the weather data with an automatic download of weather data using Weatherbit APIs.

The ML models storage consists of pickle files containing the ML models. MLflow was thought of as a possible ML model storage in the complete system integration for managing the ML models.

[8] `https://pandas.pydata.org/`
[9] `https://www.influxdata.com/`
[10] `https://mesonet.agron.iastate.edu/request/download.phtml`
[11] `https://www.weatherbit.io/`

The forecasts storage consists of CSV files containing the forecasts, which are stored using the pandas library. InfluxDB was thought of as a possible database in the complete system integration for managing the forecasts.

4.2 Electricity demand forecasting models

The electricity demand forecasting models rely on dedicated data preprocessing which consists of parsing the aggregated consumption data over the customers from a CSV file using the pandas library. Then, this basic data is enhanced with the air temperature, the apparent temperature, and the relative humidity parsed from the weather CSV file, in which the reported value is the average of the value in the last hour. Finally, data is cleaned up by filling in the gaps using the linear interpolation provided by the pandas library, which proved to be effective since there was just some missing data in the weather information. Two possible granularities are considered: hourly and daily. The data is with an hourly granularity, so for the daily granularity, data is also aggregated over the day summing the active incomings of the single hours, and averaging the weather data over the day.

The developed models are some baseline approaches that consider repeating past days and weeks, a SARIMA model, a support vector regressor model, a hist gradient boosting regressor model, an extreme gradient boosting regressor model, a prophet model, a LSTM model, a GRU model, a CNN model, some models derived by the combination of the previous techniques, a TFT model, and an AutoML approach.

The baseline approaches are developed using the NumPy library[12]. In particular, the tile method is used in different ways depending on the considered baseline. For the one-day baseline, the last day of the training active incomings is replicated for the days of the test set. For the one-week baseline, the last week of the training active incomings is replicated for the weeks of the test set.

The SARIMA model is developed using the pmdarima library[13]. The auto_arima method is used to automatically discover the optimal order for the SARIMA model to better fit the training active incomings using the week as the period for seasonal differencing and in case of the hourly granularity this is done for each hour. For the prediction, it simply takes in input the length of the test set and computes the predictions.

Support vector regressor and hist gradient boosting regressor models are developed using the scikit-learn library[14]. The extreme gradient boosting regressor model is developed using the XGBoost library[15]. Support vector regressor, hist gradient boosting regressor, and extreme gradient boosting regressor models take advantage of the following features for the hourly granularity: the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, and the relative humidity in the considered hour. In addition, the active incomings of the past 24 hours and the ones of the same hour considered in the previous 14 days are also used. For the daily granularity instead, the hour information of the date for which to compute the prediction is not considered as a feature, the weather information is obtained as the average over the day, and in place of the active incomings described for the hourly granularity, the daily active incomings of the past 14 days are used. The training is performed using the fit method on the training data for one-step ahead forecasting. The specific models' parameters are discussed in the dedicated section of chapter 5. For the prediction, an recursive strategy is used by forecasting one time instant at a time using weather forecasts, and then uaing the predicted demand data as previous information for the next time instants.

The prophet model is developed using the prophet library[16]. The fit method tasks as input a pandas data frame with just the information related to the time instants and the training active incomings and fits the optimal prophet model. For the prediction, it takes as input a pandas data frame with the time instants for which to compute the predictions and computes them, providing also uncertainty intervals.

[12] `https://numpy.org/`
[13] `http://alkaline-ml.com/pmdarima/`
[14] `https://scikit-learn.org/`
[15] `https://xgboost.readthedocs.io/`
[16] `https://facebook.github.io/prophet/`

LSTM, GRU, and CNN models are developed using the Keras library[17]. These models take advantage of the following features for the hourly granularity: the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, and the relative humidity in the considered hour. In addition, the active incoming of the previous hour is also used. Since these models are able to deal with sequences, a look back at the past 3 hours and at the same hour over the past 14 days is considered and passed as input to the models with the same features as the time instant for which to compute the prediction but with the actual consumption data of the past time instants. For the daily granularity instead, the hour of the date for which to compute the prediction is not considered as a feature, the weather information is obtained as the average over the day, and the considered look back is at the past 14 days. The training is performed using the fit method on the training data for one-step ahead forecasting. The specific models' architecture and parameters are discussed in the dedicated section of chapter 5. For the prediction, an recursive strategy is used by forecasting one time instant at a time using weather forecasts, and then uaing the predicted demand data as previous information for the next time instants.

The models derived by the combination of the previous techniques consist of the combination of one-day baseline, one-week baseline, and prophet with possibly one of LSTM, GRU, or CNN. Further details of the optimal combinations are discussed in the dedicated section of chapter 5.

The TFT model is developed using the PyTorch Forecasting library[18]. A TimeSeriesDataSet is constructed using the following features for the hourly granularity: the time index, the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, and the relative humidity in the considered hour. For the daily granularity instead, the hour of the date for which to compute the prediction is not considered as a feature, and the weather information is obtained as the average over the day. The training is performed using the fit method of the Trainer object provided by the PyTorch Lightning library[19] on the training data. The specific models' parameters are discussed in the dedicated section of chapter 5. For the prediction, it takes as input a TimeSeriesDataSet with the time instants for which to compute the predictions and computes them.

The AutoML approach is based on the TimeSeriesForecastingTask of the Auto-PyTorch library[20]. The AutoML approach takes advantage of the following features for the hourly granularity: the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, and the relative humidity in the considered hour. For the daily granularity instead, the hour of the date for which to compute the prediction is not considered as a feature, and the weather information is obtained as the average over the day. The search method is used to search for the best pipeline configuration for the given training data optimizing the machine learning models and building an ensemble out of them. For the prediction, it takes as input a pandas data frame with the time instants for which to compute the predictions and computes them.

4.3 Consumption baseline forecasting models

The consumption baseline forecasting models rely on dedicated data preprocessing which consists of parsing single customer consumption data from a CSV file using the pandas library. Then, this basic data is enhanced with the air temperature, the apparent temperature, and the relative humidity parsed from the weather CSV file, in which the reported value is the average of the value in the last hour. Finally, data is cleaned up by filling in the gaps using the linear interpolation provided by the pandas library, which proved to be effective since there was just some missing data in the weather information. Three possible granularities are considered: hourly, daily, and by tariff. The data is with an hourly granularity, so for the daily granularity, data is also aggregated over the day summing the active incomings of the single hours, and averaging the weather data over the day. The other possible

[17] `https://keras.io/`
[18] `https://pytorch-forecasting.readthedocs.io/`
[19] `https://www.pytorchlightning.ai/index.html`
[20] `https://automl.github.io/Auto-PyTorch/master/`

aggregation is by tariff, in fact, the hourly tariff proposed by MIWEnergia is different based on the day and the hour of the day. For the considered residential users the possible tariffs are 3:

1. Working days from 10 AM to 2 PM, and from 6 PM to 10 PM;
2. Working days from 8 AM to 10 AM, from 2 PM to 6 PM, and from 10 PM to midnight;
3. Saturday, Sunday, National Holidays, and working days from midnight to 8 AM.

For the tariff granularity, data is aggregated over the day summing the active incomings of the single hours for each tariff, and averaging the weather data over the tariff ranges.

The developed models are some baseline approaches that consider repeating past days and weeks, a SARIMA model, a support vector regressor model, a hist gradient boosting regressor model, an extreme gradient boosting regressor model, a prophet model, a LSTM model, a GRU model, a CNN model, a TFT model, and an AutoML approach.

The baseline approaches are developed using the NumPy library. In particular, the tile method is used in different ways depending on the considered baseline. For the one-day baseline, the last day of the training active incomings is replicated for the days of the test set. For the one-week baseline, the last week of the training active incomings is replicated for the weeks of the test set. The same was also done considering the average of the previous 4 weeks and of the previous 12 weeks of the training active incomings.

The SARIMA model is developed using the pmdarima library. The auto_arima method is used to automatically discover the optimal order for the SARIMA model to better fit the training active incomings using the week as the period for seasonal differencing and in case of the hourly granularity this is done for each hour. For the prediction, it simply takes in input the length of the test set and computes the predictions.

Support vector regressor and hist gradient boosting regressor models are developed using the scikit-learn library. The extreme gradient boosting regressor model is developed using the XGBoost library. Support vector regressor, hist gradient boosting regressor, and extreme gradient boosting regressor models take advantage of the following features for the hourly granularity: the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, and the relative humidity in the considered hour. In addition, the active incomings of the past 24 hours and the ones of the same hour considered in the previous 14 days are also used. For the daily and tariff granularities instead, the hour information of the date for which to compute the prediction is not considered as a feature, the weather information is obtained as the average over the day or tariff ranges, and in place of the active incomings described for the hourly granularity, the daily or tariff active incomings of the past 14 days are used. The training is performed using the fit method on the training data for one-step ahead forecasting. The specific models' parameters are discussed in the dedicated section of chapter 5. For the prediction, an recursive strategy is used by forecasting one time instant at a time using weather forecasts, and then uaing the predicted demand data as previous information for the next time instants.

The prophet model is developed using the prophet library. The fit method tasks as input a pandas data frame with just the information related to the time instants and the training active incomings and fits the optimal prophet model. For the prediction, it takes as input a pandas data frame with the time instants for which to compute the predictions and computes them, providing also uncertainty intervals.

LSTM, GRU, and CNN models are developed using the Keras library. These models take advantage of the following features for the hourly granularity: the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, and the relative humidity in the considered hour. In addition, the active incoming of the previous hour is also used. Since these models are able to deal with sequences, a look back at the past hour and at the same hour over the past 14 days is considered and passed as input to the models with the same features as the time instant for which to compute the prediction but with the actual consumption data of the past time instants. For the daily and tariff granularities instead, the hour of the date for which to compute the prediction is not considered as a feature, the weather

information is obtained as the average over the day or tariff ranges, and the considered look back is at the past 14 days. The training is performed using the fit method on the training data for one-step ahead forecasting. The specific models' architecture and parameters are discussed in the dedicated section of chapter 5. For the prediction, an recursive strategy is used by forecasting one time instant at a time using weather forecasts, and then uaing the predicted demand data as previous information for the next time instants.

The TFT model is developed using the PyTorch Forecasting library. A TimeSeriesDataSet is constructed using the following features for the hourly granularity: the time index, the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, and the relative humidity in the considered hour. For the daily and tariff granularities instead, the hour of the date for which to compute the prediction is not considered as a feature, and the weather information is obtained as the average over the day or tariff ranges. The training is performed using the fit method of the Trainer object provided by the PyTorch Lightning library on the training data. The specific models' parameters are discussed in the dedicated section of chapter 5. For the prediction, it takes as input a TimeSeriesDataSet with the time instants for which to compute the predictions and computes them.

The AutoML approach is based on the TimeSeriesForecastingTask of the Auto-PyTorch library. The AutoML approach takes advantage of the following features for the hourly granularity: the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, and the relative humidity in the considered hour. For the daily and tariff granularities granularity instead, the hour of the date for which to compute the prediction is not considered as a feature, and the weather information is obtained as the average over the day or tariff ranges. The search method is used to search for the best pipeline configuration for the given training data optimizing the machine learning models and building an ensemble out of them. For the prediction, it takes as input a pandas data frame with the time instants for which to compute the predictions and computes them.

4.4 Electricity production forecasting models

The electricity production forecasting models rely on dedicated data preprocessing which consists of parsing single PV plant production data from a CSV file using the pandas library and aggregating the single PV plant data to obtain the aggregated production data over the PV plants. Then, this basic data is enhanced with the air temperature, the apparent temperature, the relative humidity, the wind speed, the wind direction, the pressure altimeter, the visibility, the sky coverage, the diffuse horizontal irradiance, the direct normal irradiance, the global horizontal irradiance, the solar radiation, the UV index, the solar elevation angle, and the solar azimuth angle parsed from the weather JSON file, in which the reported value is the average of the value in the last hour. Finally, data is cleaned up by filling in the gaps using the linear interpolation provided by the pandas library, which proved to be effective since there was just some missing data in the weather information. Only the hourly granularity is considered since PV plants are highly correlated with weather data and the aggregation over the day loses part of this correlation. The data is with an hourly granularity and the target of the predictions is the mean percentage of production, which is calculated as the division of the total produced energy by the total power of the PV plants. This allows to have a bounded value from 0 to 100 from which it is possible to obtain the total produced energy simply by multiplying it by the total power of the PV plants.

The developed models are a baseline approach that considers repeating past days, an ARIMA model, a support vector regressor model, a hist gradient boosting regressor model, an extreme gradient boosting regressor model, a prophet model, a LSTM model, a GRU model, a CNN model, some models derived by the combination of the previous techniques, a TFT model, and an AutoML approach.

The baseline approach is developed using the NumPy library. In particular, the tile method is used. For the one-day baseline, the last day of the training mean percentages of production is replicated for the days of the test set.

The ARIMA model is developed using the pmdarima library. The auto_arima method is used to automatically discover the optimal order for the ARIMA model to better fit the training mean

percentages of production and this is done for each hour. For the prediction, it simply takes in input the length of the test set and computes the predictions.

Support vector regressor and hist gradient boosting regressor models are developed using the scikit-learn library. The extreme gradient boosting regressor model is developed using the XGBoost library. Support vector regressor, hist gradient boosting regressor, and extreme gradient boosting regressor models take advantage of the following features: the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, the relative humidity, the diffuse horizontal irradiance, the direct normal irradiance, the global horizontal irradiance, the solar radiation, the UV index, and the solar elevation angle in the considered hour. In addition, the mean percentages of production of the past 24 hours and the one of the same hour considered in the previous 14 days are also used. The training is performed using the fit method on the training data for one-step ahead forecasting. The specific models' parameters are discussed in the dedicated section of chapter 5. For the prediction, an recursive strategy is used by forecasting one time instant at a time using weather forecasts, and then uaing the predicted production data as previous information for the next time instants.

The prophet model is developed using the prophet library. The fit method tasks as input a pandas data frame with just the information related to the time instants and the training mean percentages of production and fits the optimal prophet model. For the prediction, it takes as input a pandas data frame with the time instants for which to compute the predictions and computes them, providing also uncertainty intervals.

LSTM, GRU, and CNN models are developed using the Keras library. These models take advantage of the following features: the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, the relative humidity, the diffuse horizontal irradiance, the direct normal irradiance, the global horizontal irradiance, the solar radiation, the UV index, and the solar elevation angle in the considered hour. In addition, the mean percentage of production of the previous hour is also used. Since these models are able to deal with sequences, a look back at the past hour and at the same hour over the past 14 days is considered and passed as input to the models with the same features as the time instant for which to compute the prediction but with the actual production data of the past time instants. The training is performed using the fit method on the training data for one-step ahead forecasting. The specific models' architecture and parameters are discussed in the dedicated section of chapter 5. For the prediction, an recursive strategy is used by forecasting one time instant at a time using weather forecasts, and then uaing the predicted production data as previous information for the next time instants.

The models derived by the combination of the previous techniques consist of the combination of one-day baseline and prophet with possibly one of LSTM, GRU, or CNN. Further details of the optimal combinations are discussed in the dedicated section of chapter 5.

The TFT model is developed using the PyTorch Forecasting library. A TimeSeriesDataSet is constructed using the following features: the time index, the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, the relative humidity, the diffuse horizontal irradiance, the direct normal irradiance, the global horizontal irradiance, the solar radiation, the UV index, and the solar elevation angle in the considered hour. The training is performed using the fit method of the Trainer object provided by the PyTorch Lightning library[21] on the training data. The specific models' parameters are discussed in the dedicated section of chapter 5. For the prediction, it takes as input a TimeSeriesDataSet with the time instants for which to compute the predictions and computes them.

The AutoML approach is based on the TimeSeriesForecastingTask of the Auto-PyTorch library. The AutoML approach takes advantage of the following features: the hour, the day, the weekday, the month, and the year information of the time instant for which to compute the prediction, the air temperature, the apparent temperature, the relative humidity, the diffuse horizontal irradiance, the direct normal irradiance, the global horizontal irradiance, the solar radiation, the UV index, and the solar elevation angle in the considered hour. The search method is used to search for the best

[21] `https://www.pytorchlightning.ai/index.html`

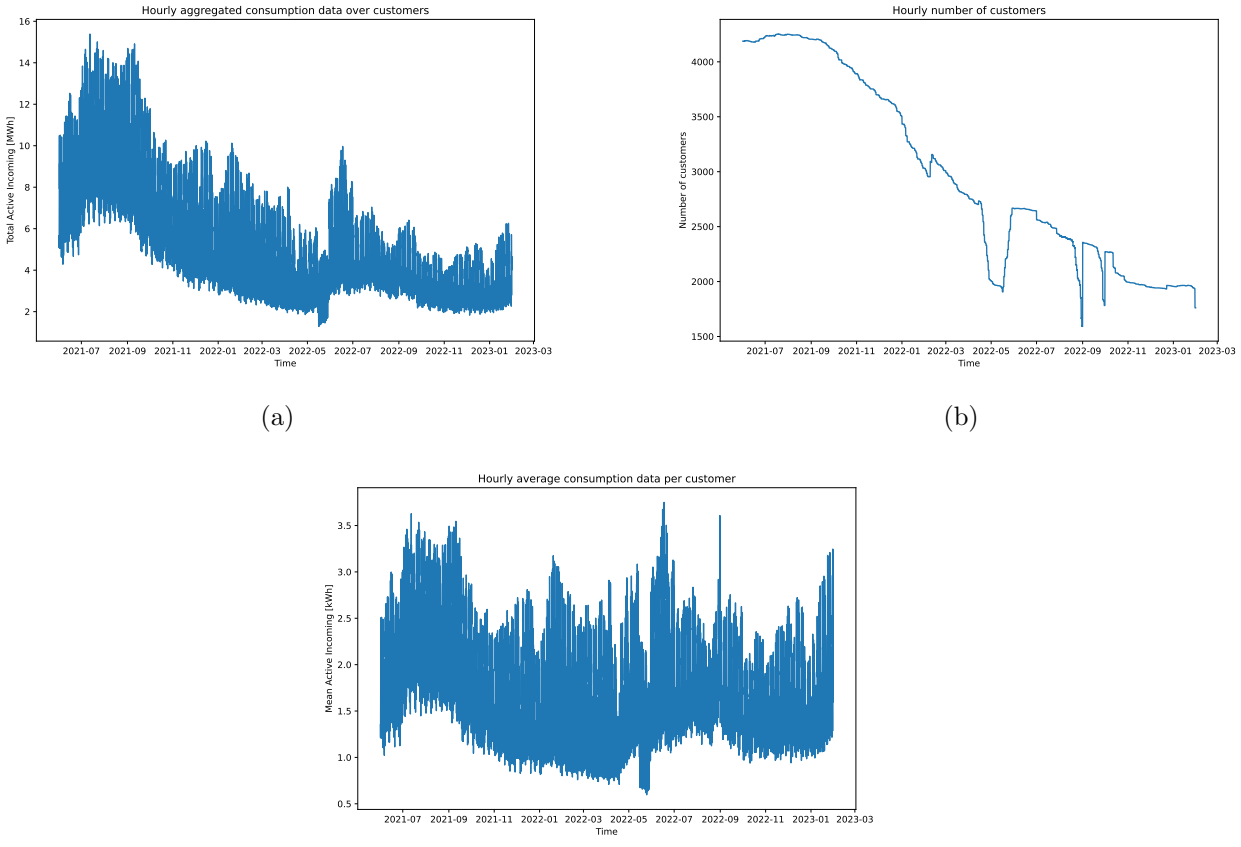
pipeline configuration for the given training data optimizing the machine learning models and building an ensemble out of them. For the prediction, it takes as input a pandas data frame with the time instants for which to compute the predictions and computes them.

5 Performance Evaluation

In this chapter, the proposed system is validated and the performance of the models for the different use cases is evaluated. The first section presents the datasets provided by MIWEnergia[22]. Subsequently, the adapted evaluation methodology is described. Finally, the evaluation of the performance of the models for the different use cases is presented. After this chapter, it will be clear how the system has been validated and what the performance achieved by the proposed system is.

5.1 MIWEnergia datasets

In this section, the MIWEnergia datasets are described. They provided 3 kinds of datasets: aggregated consumption data from all their customers, consumption data from single customers, and production data from PV plants.



(a) (b)

(c)

Figure 5.1: The graphical representation of the hourly (a) aggregated consumption over customers, (b) number of customers, and (c) average consumption per customer.

The aggregated consumption data from all their customers consists of hourly aggregated consumption data from June 2021 to January 2023 for a total of 14617 entries. The graphical representation of the hourly aggregated consumption data is reported in figure 5.1a. The number of customers is

[22] https://www.miwenergia.com/

variable, with a maximum of 4253, a minimum of 1591, a mean of 2988, and a standard deviation of 855. The graphical representation of the hourly number of customers is reported in figure 5.1b. It was thought to normalize the consumption on the basis of the number of customers in order to study the average consumption per user, as illustrated in figure 5.1c, and then multiply by the number of customers. However, this was not feasible since often this value changes significantly without reflecting on the consumption data, this information was deemed unreliable and not utilized. Despite this limitation, the average consumption per customer still provided valuable insights into consumption patterns. Specifically, it suggested the presence of two consumption peaks: one in the summer, likely attributed to air conditioning systems, and a second peak in the winter, likely caused by heating systems.

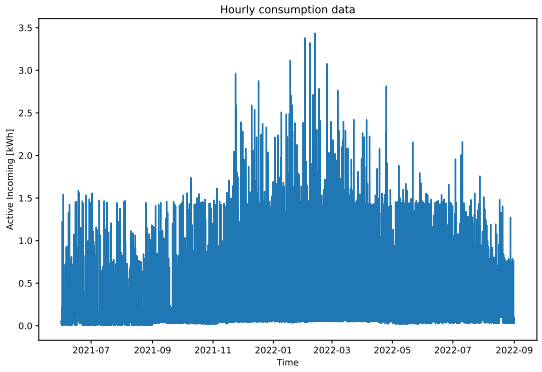
The consumption data from single customers consists of hourly aggregated consumption data of three customers:

1. from June 2021 to August 2022 for a total of 10952 total entries and it is represented in figure 5.2a;

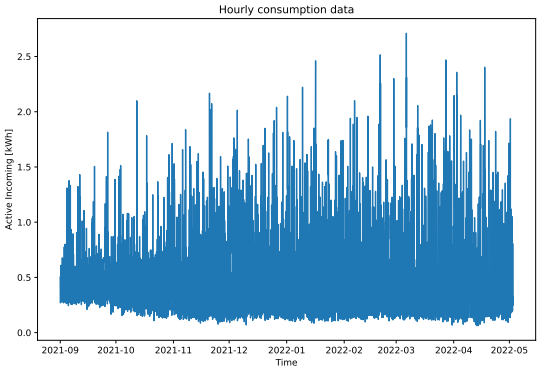
2. from September 2021 to May 2022 for a total of 5855 total entries and it is represented in figure 5.2b;

3. from September 2021 to August 2022 for a total of 8760 total entries and it is represented in figure 5.2c.

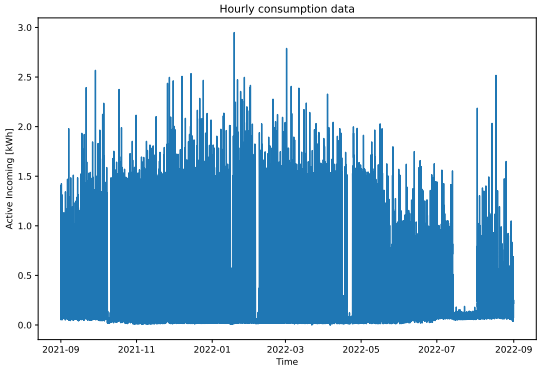
The total entries for the three customers are 25567 in the overall period from June 2021 to August 2022.



(a) First customer.



(b) Second customer.



(c) Third customer.

Figure 5.2: The graphical representation of the consumption data of the three customers.

Only consumption data for three customers were provided, as electricity consumption data of customers is considered personal data as stated in the Directive (EU) 2019/944 of the European

Parliament[23], which establishes common rules for the internal market for electricity. As such, the data falls under the scope of the General Data Protection Regulation (GDPR)[24], which requires explicit consent for processing personal data. Therefore, it is likely that the provided data came from MIWEnergia's internal employees who consented to participate in the research. In fact, the provided data were needed for a technical feasibility study. However, to identify consumption patterns among a broader population, data from more customers would be required.

By analyzing the data plots, it can be observed that the consumption patterns of the three customers are quite distinct. For instance:

- The first customer's consumption is consistently below 1.5 kWh throughout the year except for the winter season;
- The second customer shows almost the same consumption pattern;
- The third customer has a dense series with some gaps with very low consumption in parts of the year, probably due to time periods away from home.

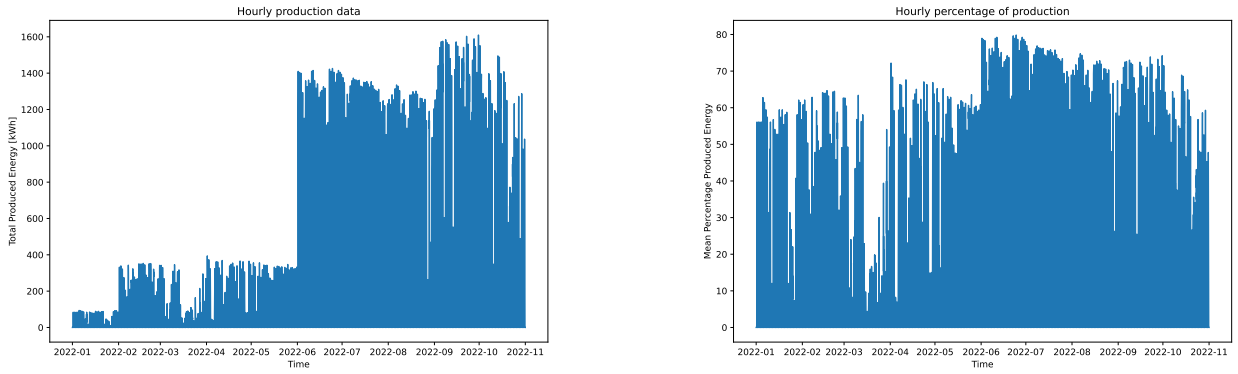
The production data from 8 PV plants consists of hourly aggregated production data:

1. from January 2022 to October 2022 for a total of 7296 total entries and it has a nominal power of 149.75 kW;
2. from February 2022 to October 2022 for a total of 6552 total entries and it has a nominal power of 237.6 kW;
3. from February 2022 to October 2022 for a total of 6552 total entries and it has a nominal power of 158.4 kW;
4. from June 2022 to October 2022 for a total of 3576 total entries and it has a nominal power of 1240 kW;
5. from September 2022 to October 2022 for a total of 1465 total entries and it has a nominal power of 126.2 kW;
6. from September 2022 to October 2022 for a total of 1465 total entries and it has a nominal power of 113 kW;
7. from September 2022 to October 2022 for a total of 1465 total entries and it has a nominal power of 45 kW;
8. from September 2022 to October 2022 for a total of 1465 total entries and it has a nominal power of 100 kW;

The total entries for the 8 PV plants are 29836 total entries in the overall period from January 2022 to October 2022, aggregating over plants the resulting entries are 7296. The graphical representation of the hourly aggregated total and mean percentage production data are reported respectively in figure 5.3a and figure 5.3b.

The same weather data from the same provider and the same weather station were used for all the tasks. While it is reasonable to assume that some weather conditions, such as clouds, may vary and affect consumption and production differently, this approach was taken since the customers and the PV plants were in the same area, with a maximum distance of around 100 km between them, and also because the value of the weather parameters is an average of the values recorded in the past hour. Therefore, using the same weather data is deemed appropriate for the analysis at hand, given that it reflects the weather conditions in the area over a broad time frame.

[23] `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32019L0944`
[24] `https://gdpr-info.eu/`



(a) (b)

Figure 5.3: The graphical representation of the hourly aggregated (a) total and (b) mean percentage production data.

5.2 Evaluation methodology

The evaluation methodology was based on two relevant error metrics: Mean Absolute Percentage Error (MAPE) and Mean Absolute Error (MAE). These are standard and widely used metrics in time series forecasting for different use cases, as reported in many articles and books such as [10, 27, 52]. The MAPE is defined as $\text{MAPE}(y, \hat{y}) = \frac{100\%}{N} \sum_{i=0}^{N-1} \frac{|y_i - \hat{y}_i|}{|y_i|}$. It is the most relevant error metric for all the tasks since it is a percentage-based error metric that takes into account the magnitude of the errors relative to the actual values. The MAE is defined as $\text{MAE}(y, \hat{y}) = \frac{\sum_{i=0}^{N-1} |y_i - \hat{y}_i|}{N}$. It is the most suitable error metric in consumption baseline forecasting where there is a high variability from very low to high values, and electricity production forecasting where there is a significant number of zeros when the sun is absent.

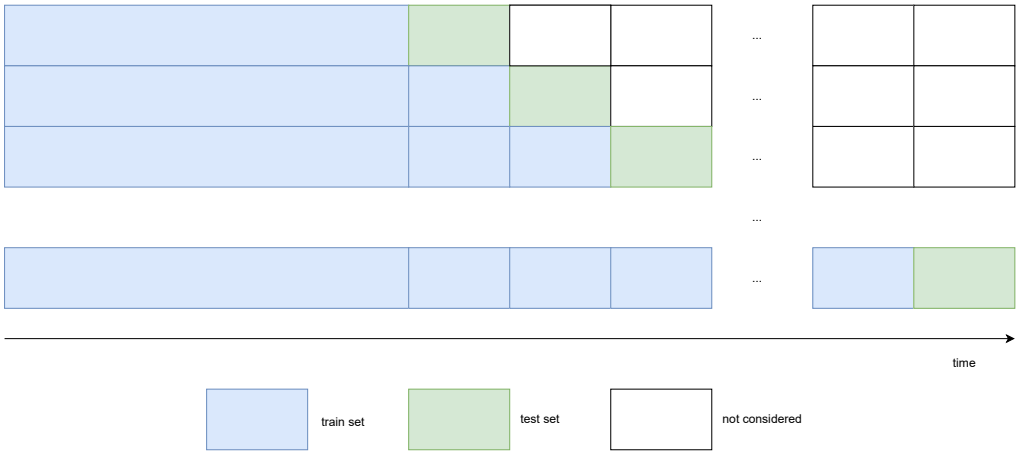


Figure 5.4: The schematic representation of the blocked k-fold cross-validation adopted.

For assessing the model performance when dealing with time series data, the traditional cross-validation techniques are not suitable as they assume that the data points are independent and identically distributed (i.i.d.), which is not the case in time series data, as also reported in the chapter 2 by the following articles [15, 20]. In fact, in time series data the order of the observations matters, and there are temporal dependencies between the observations. Therefore, a more appropriate technique for evaluating time series models is blocked k-fold cross-validation. The basic idea of blocked k-fold cross-validation is to split the data considering multiple training and test sets, where the training set only includes data from the past and the test set includes data from the future. This simulates the real-world scenario where we want to make predictions about the future based on past data. In figure 5.4, the schematic representation of how the blocked k-fold cross-validation was performed is

reported, 12 splits were used with a test size depending on the specific use case. The training set is increased by adding the previous test elements at every successive evaluation.

Another technique, reported in [20] is the repeated Holdout Out-of-sample tested in multiple testing periods with a Monte Carlo simulation using 70% of the total observations of the time series in each test. For each period, a random point is picked from the time series. The previous window comprising 60% of the time series is used for training and the following window of 10% of the time series is used for testing. In the paper, it was stated that the approach provided the most accurate estimates when the time series are non-stationary. However, for having the same evaluation mechanisms on all the models and simulating the fact that the model starts with limited data and then the training amount increases over time to obtain better performance, block validation was used using the TimeSeriesSplit provided by the scikit-learn library. Moreover, the considered data are without a strong trend as reported in the dedicated sections of the specific use cases where the data and the model forecasts are analyzed.

In addition to the blocked k-fold cross-validation results also the results on the last test split using the rest as training are reported. This was done since it provides insight on which could be the performance of the models in forecasting the near future data with the currently available training data. A table summarizing the metrics and the evaluation mechanisms adopted for the use cases is reported in table 5.1.

Use case	**Metrics**	**Evaluation mechanisms**
Electricity demand forecasting	MAPE	blocked k-fold cross-validation and test on the last split
Electricity production forecasting	MAE	blocked k-fold cross-validation and test on the last split
Consumption baseline forecasting	MAPE and MAE	blocked k-fold cross-validation and test on the last split

Table 5.1: Table summarizing the metrics and the evaluation mechanisms adopted for the use cases.

This evaluation methodology was used at training time to study the performance over time. At forecasting time, the performance on the last time slot was treated with more attention as the most relevant for the energy retailers since the most related to the performance on imminent future data. The combinations of the models were done just at forecasting time on the last time slot using the average of some approaches for studying whether there could be a beneficial effect on the forecasts and compensation of error between the different approaches. The same was also done for the AutoML approach, in fact, it was tested just at forecasting time on the last time slot for comparison to the tested models. It was included as a baseline to show how a general-purpose framework can perform in specific use cases compared to specific architectures designed for the task.

As explained in chapter 4, for all the use cases, there is a difference between training and forecasting time for the ML (support vector regressor, hist gradient boosting regressor, and extreme gradient boosting regressor) and DL (LSTM, GRU, and CNN) models. In particular, for these models, a recursive strategy to generate multi-step forecasts was adopted. This approach is a standard classical technique covered in various textbooks, including [73]. It offers advantages such as simplicity in understanding, clear temporal dependency as previous forecasts are used as input for subsequent ones, flexibility to be applied to different models, and adaptability to various forecasting horizons, which is crucial when integrating it into a SaaS solution. However, there are also disadvantages associated with this approach. Error propagation can occur, and this can be observed in the forecasting results, where small inaccuracies in initial predictions can accumulate and lead to larger errors over time. Sub-optimality is another drawback, as the optimality achieved for one-step forecasting may not generalize well to multi-step forecasting. Additionally, the computational complexity is high as the model needs to be run for each individual step. In contrast, the other approaches: baselines, ARIMA/SARIMA, Prophet, and TFT directly handle the concept of multi-step prediction and present no difference between training and forecasting time.

5.3 Electricity demand forecasting

The aggregated consumption data over the customers is analyzed to get some descriptive analytics before finding adequate models to forecast the demand. The time series decomposition using an additive model of the hourly aggregated consumption over the customers considering as period of the time series a week is reported in figure 5.5. It showed a considerable amount of noise, comparable to seasonality in magnitude. The trend showed 2 peaks during the summer season, the first being more emphasized since also corresponds to a peak in the number of users.

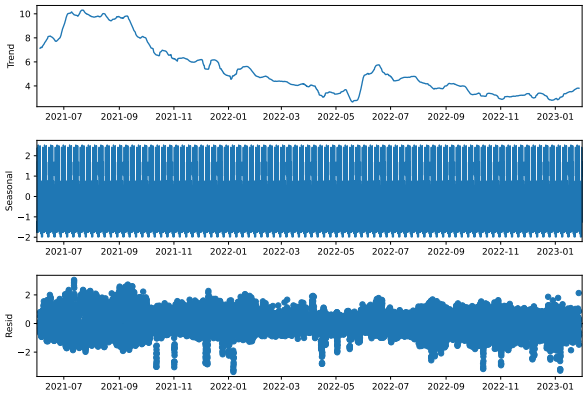


Figure 5.5: The time series decomposition of the hourly aggregated consumption over the customers considering as period of the time series a week.

The auto-correlation of the hourly aggregated consumption over the customers is reported in figure 5.6. It shows a high auto-correlation value in the close time lags and also at every 24 hours, along with an even higher value at a one-week distance. This indicates that the consumption data from the closest time lags, particularly up to three closest ones, as well as those corresponding to the same hour in the preceding days and even better in the preceding weeks, may be valuable features for predicting a time instant's demand. A reasonable balance can be achieved by incorporating the consumption data from the past 14 days.

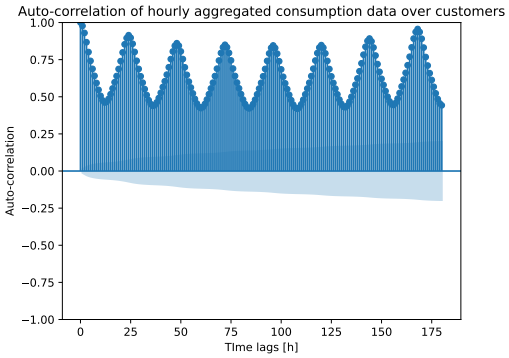


Figure 5.6: The auto-correlation of the hourly aggregated consumption over the customers.

The coefficients given by the Fourier transform for the hourly aggregated consumption over the customers are reported in figure 5.7. The graphical representation shows 2 main frequencies, one representing the weekly periodicity and one the daily periodicity. Other minor peaks are present mostly at multiples of the 1/week frequency.

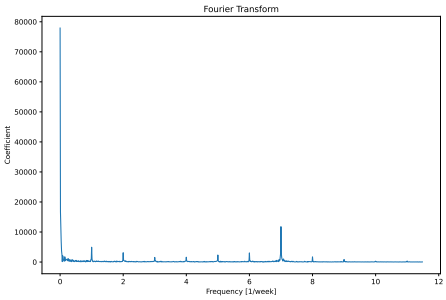


Figure 5.7: The coefficients given by the Fourier transform for the hourly aggregated consumption over the customers.

The daily aggregated consumption over the customers is reported in figure 5.8. Aggregating the data on a daily basis, it is possible to more clearly observe the weekly pattern of consumption, which exhibits an increase on weekdays and a decrease on weekends.

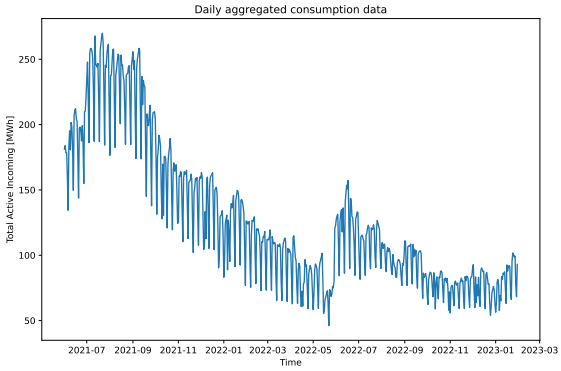


Figure 5.8: The daily aggregated consumption over the customers.

Figure 5.9 shows the time series decomposition of the daily aggregated consumption over the customers, using an additive model with a period of one week. The decomposition is consistent with the hourly aggregated consumption over the customers.

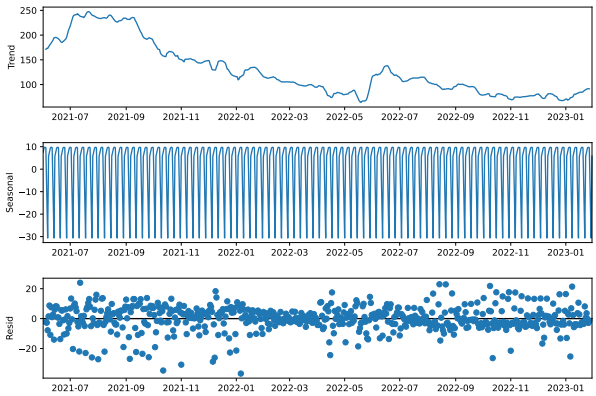


Figure 5.9: The time series decomposition of the daily aggregated consumption over the customers considering as period of the time series a week.

The auto-correlation of the daily aggregated consumption over the customers is reported in figure 5.10. It shows a high auto-correlation value in the closest time lag, with an even greater value at every week lag. This suggests that useful features, since highly correlated, for predicting a time

instant's demand may be the closest ones, but also those from the same day of the week in the previous weeks. A good trade-off can be achieved by considering the consumption data in the previous 14 days.

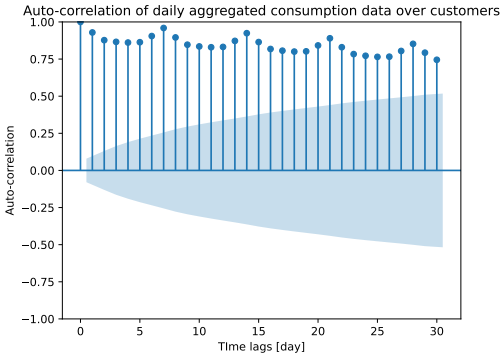


Figure 5.10: The auto-correlation of the daily aggregated consumption over the customers.

The coefficients given by the Fourier transform for the daily aggregated consumption over the customers are reported in figure 5.11. The graphical representation shows 1 main frequency representing the weekly periodicity and other minor peaks at multiples of the 1/week frequency.

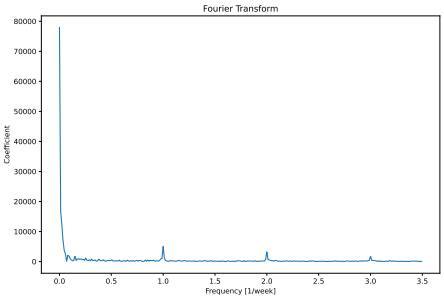


Figure 5.11: The coefficients given by the Fourier transform for the daily aggregated consumption over the customers.

Basic data is enhanced with the air temperature, the apparent temperature, and the relative humidity since they are considered the only weather features capable of influencing customers' energy consumption. To assess the relationship between these weather variables and the aggregated consumption over the customers, two correlation coefficients were used: Pearson's correlation coefficient and Spearman's rank correlation coefficient. Pearson's correlation coefficient measures the strength of the linear relationship between two variables, while Spearman's rank correlation coefficient measures the strength of the monotonic relationship. The pearsonr and spearmanr methods of the SciPy library[25] were used to compute the correlation with weather data. The results showed that the hourly aggregated consumption over the customers had:

- a Pearson correlation coefficient of 0.4480 and a Spearman's rank correlation coefficient of 0.4378 with respect to the air temperature;
- a Pearson correlation coefficient of 0.4475 and a Spearman's rank correlation coefficient of 0.4368 with respect to the apparent temperature;
- a Pearson correlation coefficient of -0.2689 and a Spearman's rank correlation coefficient of -0.3173 with respect to the relative humidity.

[25] `https://scipy.org/`

It can be noticed that both the coefficients indicate a moderate correlation between the weather variables and consumption, this suggests that incorporating weather data into the prediction models could be useful.

The results showed that the daily aggregated consumption over the customers had:

- a Pearson correlation coefficient of 0.3898 and a Spearman's rank correlation coefficient of 0.3119 with respect to the air temperature;
- a Pearson correlation coefficient of 0.3823 and a Spearman's rank correlation coefficient of 0.3114 with respect to the apparent temperature;
- a Pearson correlation coefficient of -0.0908 and a Spearman's rank correlation coefficient of -0.1547 with respect to the relative humidity.

It can be noticed that both the coefficients decreased considering the daily data meaning that the mean weather data over the day is less correlated to the daily aggregated consumption over the customers, compared to the hourly granularity.

After this data analysis, the specific parameters used in the models can be explained more in detail. The parameters for the different models were tested and verified on training data to result in the best-performing models, this was done by trying different values and configurations. The baseline approaches are built considering the repetition of past days and weeks since data presents a high correlation with that time instants and could lead to a reasonable baseline performance to achieve. The SARIMA model considers the week as the period for seasonal differencing since data presents a high correlation with that time instants and the model can try to take advantage of the weekly seasonality. The support vector regressor model uses a radial basis function kernel with a configuration of the C parameter (regularization parameter of squared l2 penalty) to 1.0 to penalize the complexity of the model, and the epsilon parameter defining the epsilon-tube for no penalty to 0.1 as a trade-off between accuracy and generalization. The hist gradient boosting regressor model uses as loss the absolute error since this was the supported metric closer to our evaluation metric. The learning rate was set to 0.1, 100 estimators were used and l2 regularization was not set. The parameters for tree definition are 31 as the maximum number of leaves, depth is not constrained, 20 as the minimum number of samples per leaf, and 255 as the maximum number of bins. For the extreme gradient boosting regressor model, tree-based models were used and the construction algorithm is automatically chosen by heuristic to choose the fastest method. It uses a squared loss since this was the supported metric closer to our evaluation metric. The learning rate was set to 0.3, 100 estimators were used and l2 regularization was set with a weight of 1.0. The parameters for tree definition are 6 as the maximum depth, and the maximum number of leaves is not constrained. The Prophet model was built keeping the default parameters for automatically detecting seasonalities and best fitting the training data.

The LSTM model was designed as a 2-layer model with 24 Bidirectional LSTM units in the first layer which use rectified linear unit (ReLU) activation function and sigmoid as recurrent activation function with both dropout and recurrent dropout of 0.02. Batch normalization is then applied before entering the second layer composed of 16 Bidirectional LSTM units which use ReLU activation function and sigmoid as recurrent activation function without dropout and recurrent dropout. The output of the layer goes inside a dense unit to output the final prediction. The model is trained using mean absolute percentage error as loss and Nadam (a version of Adam integrating the concept of Nesterov momentum) as optimizer with a learning rate of 0.005.

The GRU model was designed as a 3-layer model with 24 Bidirectional GRU units in the first layer which use ReLU activation function and sigmoid as recurrent activation function with both dropout and recurrent dropout of 0.02. Subsequently, the second layer is composed of 16 Bidirectional GRU units which use ReLU activation function and sigmoid as recurrent activation function without dropout and recurrent dropout. The output of the second layer goes inside a third dense layer composed of 4 units with ReLU activation function before entering the final dense unit to output the final prediction. The model is trained using mean absolute percentage error as loss and Nadam as optimizer with a learning rate of 0.005.

The CNN model was designed as a 3-layer model with a first layer composed of 24 1D Convolutional units with a kernel size of 5 which uses ReLU activation function. A 1D max polling operation is applied before entering the second layer composed of 16 1D Convolutional units with a kernel size of 3 which use ReLU activation function. A 1D max polling operation is applied before the flatting operation and entering the third dense layer composed of 4 units with ReLU activation function. The output of the layer goes inside a dense unit to output the final prediction. The model is trained using mean absolute percentage error as loss and Nadam as optimizer with a learning rate of 0.005.

The TFT model was configured with 2 LSTM layers with 16 as hidden size, 4 as attention size, and 8 as hidden continuous size. The dropout is set to 0.02. The model is trained using mean absolute percentage error as loss and Nadam as optimizer with a learning rate of 0.005.

12 splits were used for block validation with a test size of a month each time, namely 30 days of data. For the hourly granularity, the models start with 5977 entries as training and predict every time 720 entries until reaching the last prediction instant where the model has a training size of 13897. The results for hourly aggregation are reported in table 5.2. Entries are sorted by best MAPE using the last split as the test set. The blocked k-fold cross-validation suggests that the one-week baseline has the best MAPE, but looking at the last split this is not confirmed. This is probably due to the fact that overall the weekly repetition seems to be quite effective over the year, but there are better approaches that with the increase of training data perform better, this can be the case in GRU and CNN since the results on the last split are quite good compared to the mean MAPE obtained over the splits. This can be explained by the fact that, due to their complexity, these models require more data and in the first splits there was not even a single year of data, so they didn't even have the possibility to infer a possible annual seasonality. Results seem also to confirm that not always sophisticated models perform better and some of them are no better than a baseline. Models treated in the literature that are performing well in similar tasks can also not be suited for this specific use case. Data are a big difference and with a little amount of data for some complex networks is hard to perform well and generalize. Instead, the TFT approach seems to have pretty much the same MAPE over the considered splits confirming the power of this architecture despite the limited data In contrast, the boosters that have a good blocked k-fold MAPE on the last split do not perform well.

Model	**Blocked k-fold cross-validation MAPE**	**Test on the last split MAPE**
TFT	15.00 ± 4.31	14.76
GRU	30.77 ± 14.48	16.82
CNN	34.64 ± 15.76	17.49
One Week Baseline	12.94 ± 4.27	17.55
SARIMA	15.44 ± 4.57	18.47
One Day Baseline	22.86 ± 7.07	20.54
LSTM	34.08 ± 11.92	23.18
HistGradientBoostingRegressor	14.54 ± 5.51	29.33
SVR	61.80 ± 22.63	29.47
Prophet	28.68 ± 7.07	31.62
XGBRegressor	16.88 ± 8.08	37.81

Table 5.2: Table summarizing the results for hourly aggregation.

The results for hourly aggregation at forecasting time on the last time slot including also the combinations of different techniques and the AutoML approach are reported in table 5.3. The combination of the previous techniques was performed by combining one-day baseline, one-week baseline, and prophet with possibly one of LSTM, GRU, or CNN with the intent of trying to improve the robustness of DL models with some stable baselines and prophet model which internally automatically deals with seasonalities. The AutoML approach uses the mean MAPE forecasting as loss and it is launched for 10 hours with a maximum function evaluation time of 2 hours. Then the best ensemble of the found models is returned and can be used to forecast. In the table, it is also reported the performance on the one-week horizon, namely 7 days of data, from which it is possible to gain more insights on the models'

forecasts. Including also combinations and AutoML, there are a few observations that can be done. The combinations of DL models and the one-week baseline are the most beneficial, reducing of many point percentage the MAPE of the techniques on the one-month time horizon. This suggests that it can make sense to further investigate the concept of ensemble learning and optimize this combination at training time to possibly have stronger results at forecasting time. AutoML has great potential and obtains very good results compared to many other models. Potentially providing more total training time it can be further optimized to reach better performance. TFT still dominates on the one-month MAPE but not on the one-week one on which the one-week baseline and many combinations but also SARIMA and hist gradient boosting regressor perform better. This confirmed that the strength of TFT is to have good performance and preserve it also on larger prediction horizons.

Model	**Forecast on the last split one-week MAPE**	**Forecast on the last split one-month MAPE**
TFT	13.71	14.76
GRU + One Week Baseline	11.79	14.79
CNN + One Week Baseline	11.84	15.39
GRU	17.25	16.82
AutoML	14.68	16.96
CNN + One Week Baseline + Prophet	15.43	17.13
CNN + One Day Baseline + One Week Baseline + Prophet	13.60	17.25
GRU + One Day Baseline + One Week Baseline + Prophet	13.25	17.33
GRU + One Week Baseline + Prophet	15.63	17.37
CNN	14.59	17.49
One Week Baseline	10.87	17.55
LSTM + One Week Baseline + Prophet	13.16	17.89
LSTM + One Day Baseline + One Week Baseline + Prophet	12.58	18.10
LSTM + One Week Baseline	14.42	18.11
SARIMA	12.28	18.47
One Day Baseline	14.78	20.54
One Day Baseline + One Week Baseline + Prophet	15.15	20.97
One Week Baseline + Prophet	19.02	22.80
LSTM	20.94	23.18
HistGradientBoostingRegressor	13.00	29.33
SVR	29.06	29.47
Prophet	33.94	31.62
XGBRegressor	17.84	37.81

Table 5.3: Table summarizing the results for hourly aggregation at forecasting time.

The forecasts produced by the TFT model for the last split in comparison to the actual values are reported in figure 5.12. It can be noticed that for the first week, it follows very accurately the signal, the second and third weeks are pretty good but it is missing the peaks over the day and the last week is decreasing while the actual values have high peaks. The absolute percentage error of forecasts produced by the TFT model for the last split in comparison to the actual values is reported in figure 5.13. It can be shown that except for some peaks, the percentage error stays below 30%, with most values around 10%. Finally, the scatterplot representing the error of the forecasts produced by

the TFT model for the last split in comparison to the actual values is reported in figure 5.14. This scatterplot demonstrates that there is no bias and can be also noticed that when the predicted values are low also the error is limited, while when the prediction is high there are both high positive and negative errors.

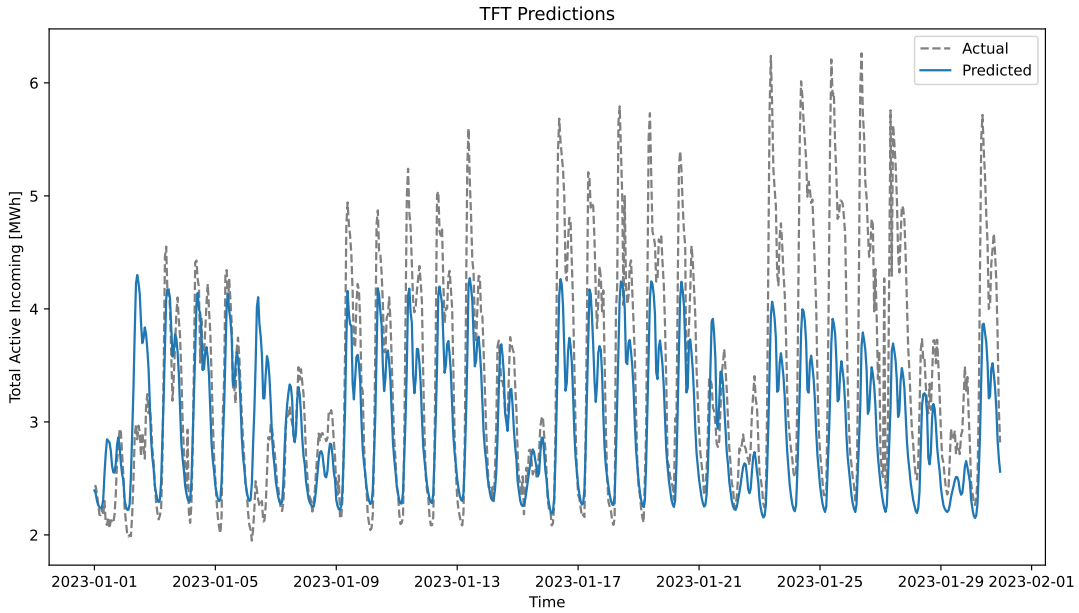


Figure 5.12: The forecasts produced by the temporal fusion transoformer model for the last split in comparison to the actual values.

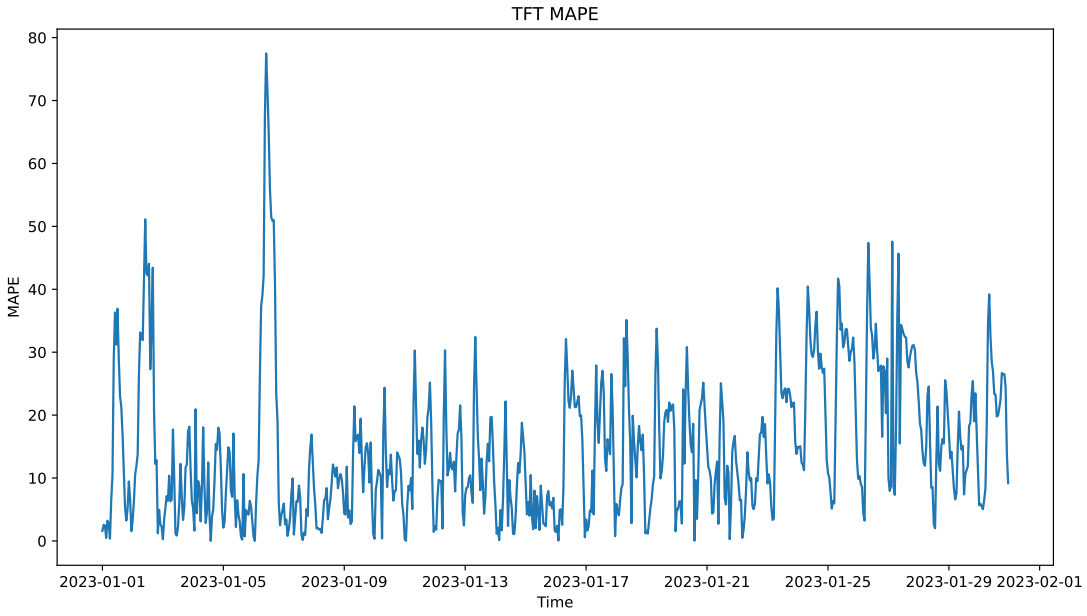


Figure 5.13: The absolute percentage error of forecasts produced by the temporal fusion transoformer model for the last split in comparison to the actual values.

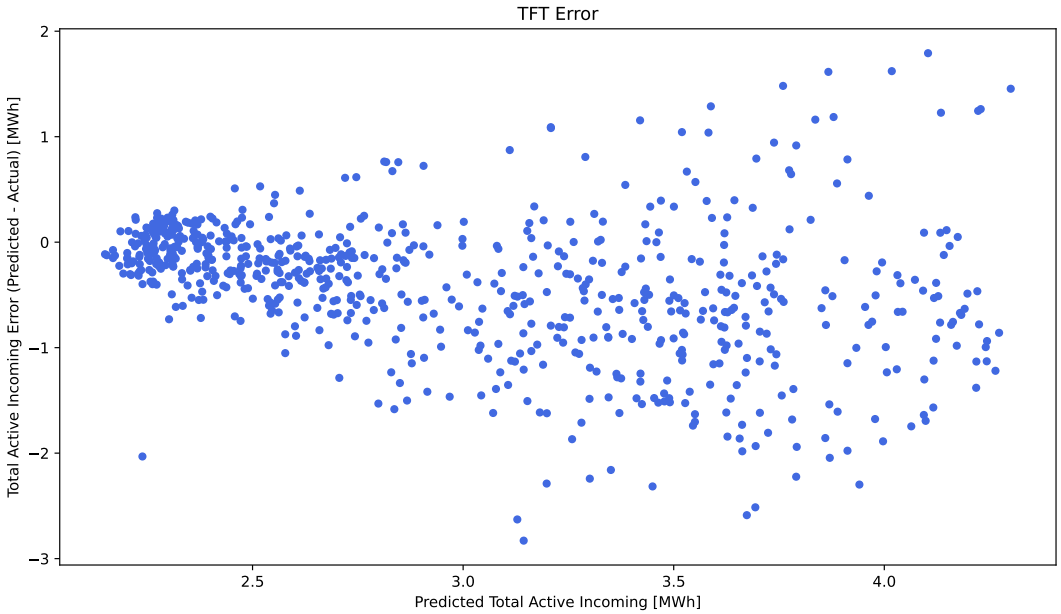


Figure 5.14: The scatterplot representing the error of the forecasts produced by the temporal fusion transoformer model for the last split in comparison to the actual values.

For the daily granularity, the models start with 249 entries as training and predict every time 30 entries until reaching the last prediction instant where the model has a training size of 579. The results for daily aggregation are reported in table 5.4. In this case, TFT is no more the best performing model, probably due to the fact of 24 times less quantity of data due to the daily aggregation. Though, it is still a well-performing model on both blocked k-fold cross-validation and last split results. LSTM and CNN perform better in terms of the last split MAPE but present a high blocked k-fold MAPE, as in the case of the hourly granularity probably that to the increase of data, the results on the last time slots improved in a significant way. Also, the boosters present very good results on the last time split and also on the blocked k-fold validation showing consistency over time and good results also with few training data.

Model	**Blocked k-fold cross-validation MAPE**	**Test on the last split MAPE**
CNN	50.55 ± 36.81	7.49
XGBRegressor	12.96 ± 4.36	7.65
HistGradientBoostingRegressor	13.02 ± 5.54	9.81
LSTM	29.60 ± 13.81	14.51
TFT	13.73 ± 5.06	14.88
One Week Baseline	11.50 ± 4.78	18.40
SARIMA	11.92 ± 7.04	19.63
GRU	28.76 ± 29.26	20.47
One Day Baseline	22.80 ± 8.21	21.51
Prophet	19.06 ± 10.81	26.33
SVR	53.63 ± 22.39	39.82

Table 5.4: Table summarizing the results for daily aggregation.

The results for daily aggregation at forecasting time on the last time slot including also the combinations of different techniques and the AutoML approach are reported in table 5.5. In this case, the impact of the combinations is not so evident and does not improve the results in all the cases. AutoML presents not very good results being worse than the one-week baseline, this is probably due

to the scarcity of training data and the fact of AutoML of trying very complex architects when also simple ones may work well in certain cases.

Model	**Forecast on the last split one-week MAPE**	**Forecast on the last split one-month MAPE**
CNN	8.36	7.49
XGBRegressor	11.78	7.65
HistGradientBoostingRegressor	12.42	9.81
CNN + One Week Baseline	8.33	12.28
LSTM + One Week Baseline	11.19	12.91
LSTM	18.21	14.51
TFT	12.65	14.88
LSTM + One Week Baseline + Prophet	8.81	15.91
CNN + One Week Baseline + Prophet	11.34	16.96
LSTM + One Day Baseline + One Week Baseline + Prophet	9.82	17.29
CNN + One Day Baseline + One Week Baseline + Prophet	10.88	17.88
One Week Baseline	9.63	18.40
GRU + One Week Baseline	9.15	19.18
SARIMA	11.23	19.63
GRU	10.87	20.47
AutoML	10.61	20.98
GRU + One Day Baseline + One Week Baseline + Prophet	11.29	21.34
One Day Baseline	13.14	21.51
GRU + One Week Baseline + Prophet	11.89	21.56
One Day Baseline + One Week Baseline + Prophet	11.72	21.69
One Week Baseline + Prophet	12.83	22.21
Prophet	17.35	26.33
SVR	58.09	39.82

Table 5.5: Table summarizing the results for hourly aggregation at forecasting time.

The forecasts produced by the CNN model for the last split in comparison to the actual values are reported in figure 5.15. It can be noticed that for the first three weeks, it follows very accurately the signal and the last week is increasing correctly but not enough as the actual values. The absolute percentage error of forecasts produced by the CNN model for the last split in comparison to the actual values is reported in figure 5.16. It can be shown that except for some peaks, the percentage error stays below 10%. Finally, the scatterplot representing the error of the forecasts produced by the CNN model for the last split in comparison to the actual values is reported in figure 5.17. This scatterplot demonstrates that there is no bias even though that seems that when predicting high values it is missing some extra quantity in the forecasts, this is probably true in the specific considered last split and in particular the last week.

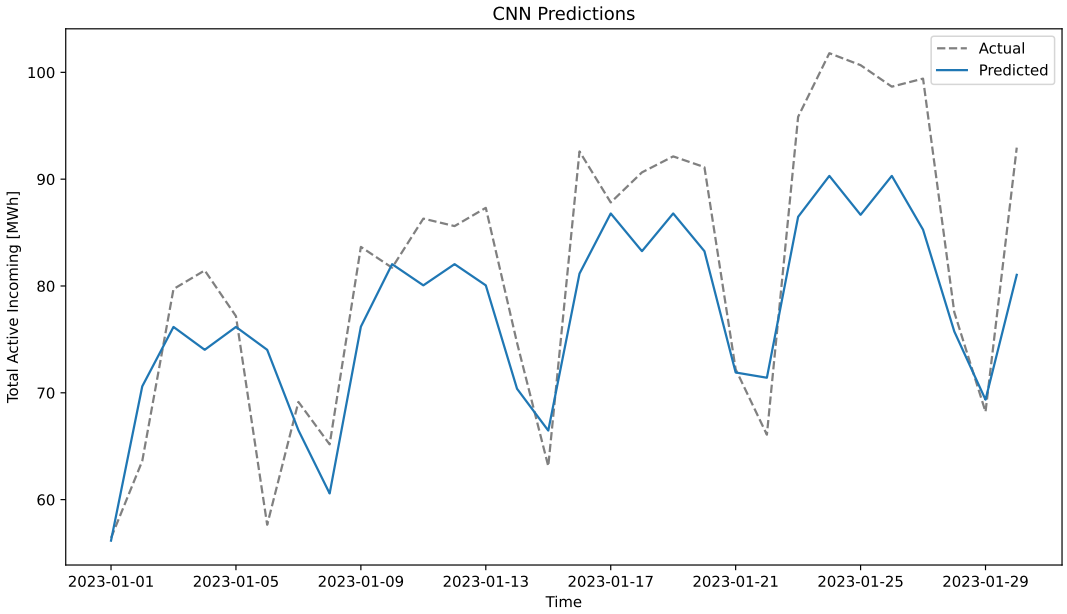


Figure 5.15: The forecasts produced by the convolutional neural network model for the last split in comparison to the actual values.

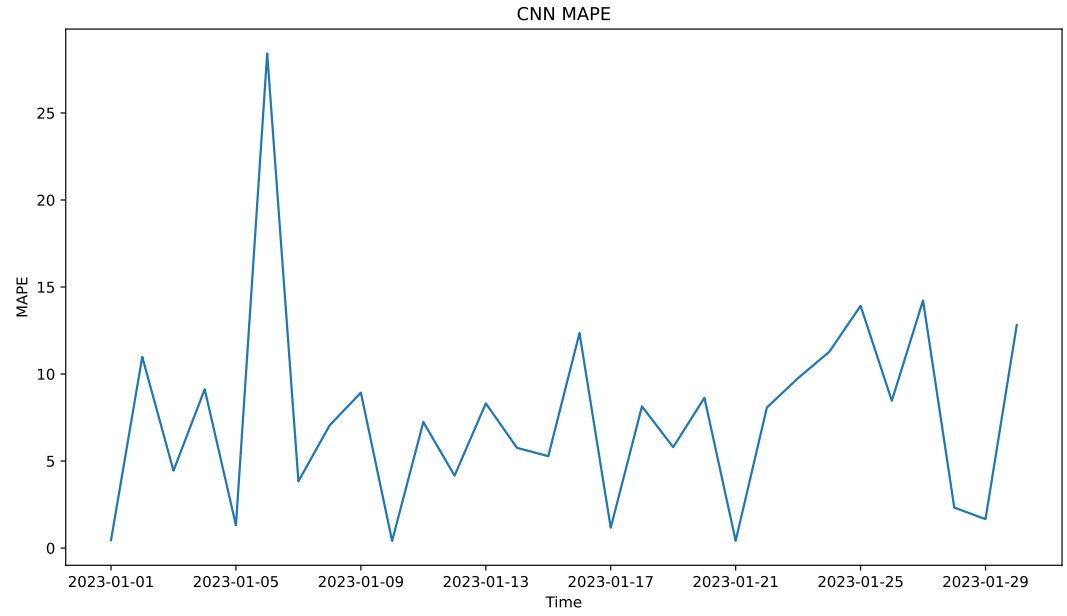


Figure 5.16: The absolute percentage error of forecasts produced by the convolutional neural network model for the last split in comparison to the actual values.

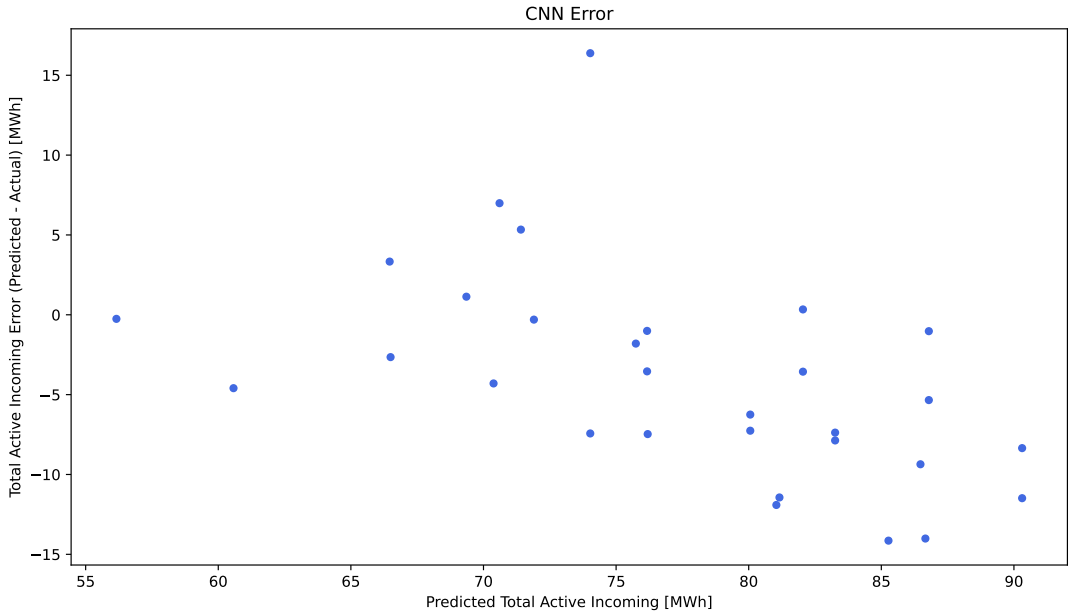


Figure 5.17: The scatterplot representing the error of the forecasts produced by the convolutional neural network model for the last split in comparison to the actual values.

5.4 Electricity production forecasting

As described in the data preprocessing in chapter 4, single PV plant production data are aggregated to obtain the aggregated production data over the PV plants. The target of the predictions is the mean percentage of production, which is calculated as the division of the total produced energy by the total power of the PV plants. This allows us to have a bounded value from 0 to 100 from which it is possible to obtain the total produced energy simply by multiplying it by the total power of the PV plants. This was also done since PV plants are added over time and this was unpredictable, this results in predicting the percentage of production of the PV plants. This data is analyzed to get some descriptive analytics before finding adequate models to forecast the production. The time series decomposition using an additive model of the hourly percentage of production considering as period of the time series a day is reported in figure 5.18. It showed a considerable amount of noise, comparable to seasonality and trend in magnitude. The trend component exhibits a clear peak during the summer season. As expected, the seasonality component appears to have the most significant impact on the time series.

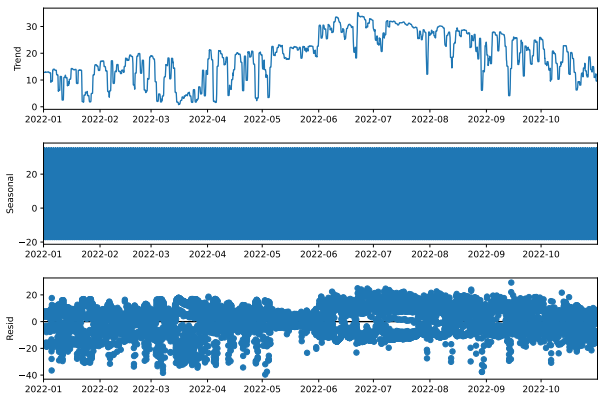


Figure 5.18: The time series decomposition of the hourly percentage of production considering as period of the time series a day.

The auto-correlation of the hourly percentage of production is reported in figure 5.19. It shows a high auto-correlation value in the closest time lag and also at every 24 hours, with the value slightly decreasing as the lag increases over days. This indicates that the production data from the closest time lag and those corresponding to the same hour in the preceding days may be valuable features for predicting a time instant's production. A reasonable balance can be achieved by incorporating the production data from the past 14 days.

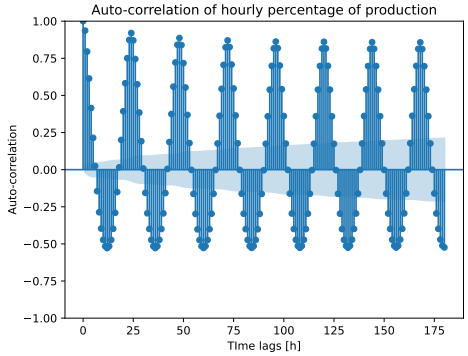


Figure 5.19: The auto-correlation of the hourly percentage of production.

The coefficients given by the Fourier transform for the hourly percentage of production are reported in figure 5.20. The graphical representation clearly shows a main frequency at the daily periodicity. Other minor peaks are present mostly at multiples of the 1/day frequency.

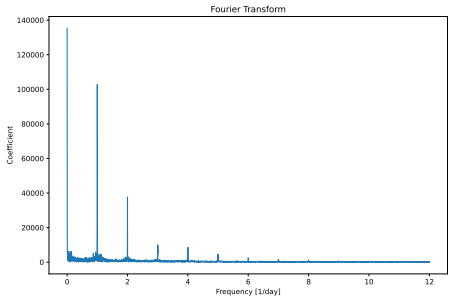


Figure 5.20: The coefficients given by the Fourier transform for the hourly percentage of production.

The daily percentage of production is reported in figure 5.21.

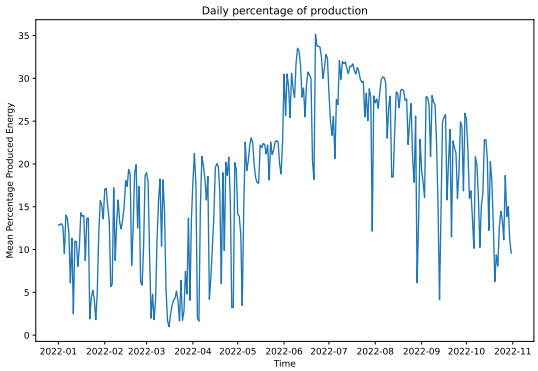


Figure 5.21: The daily percentage of production.

The time series decomposition using an additive model of the daily percentage of production considering as period of the time series a week is reported in figure 5.22. It showed a considerable

amount of noise, comparable to the trend in magnitude when not in the summer season. The trend component exhibits a clear peak during the summer season and appears to be the significant component of the time series. In this case, considering a weekly period, the seasonality component appears to not have a significant impact on the time series.

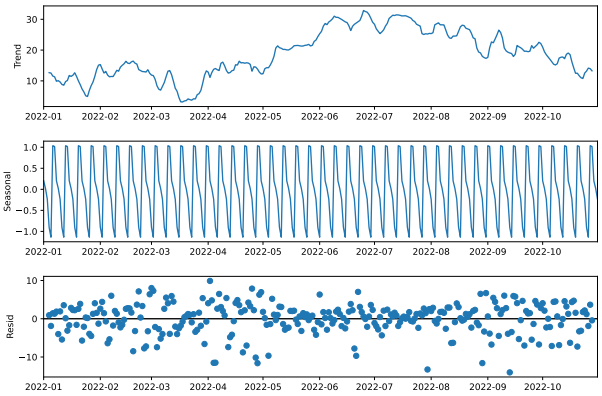


Figure 5.22: The time series decomposition of the daily percentage of production considering as period of the time series a week.

The auto-correlation of the daily percentage of production is reported in figure 5.23. It shows a high auto-correlation value in the closest time lag and then a slight decrease as the lag increases over days.

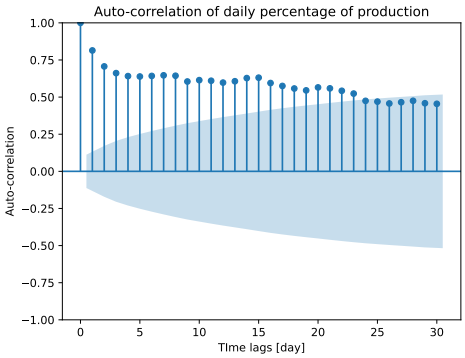


Figure 5.23: The auto-correlation of the daily percentage of production.

The coefficients given by the Fourier transform for the daily percentage of production are reported in figure 5.24. As expected from the time series decomposition, the graphical representation exhibits that there are no main frequencies in the daily percentage of production.

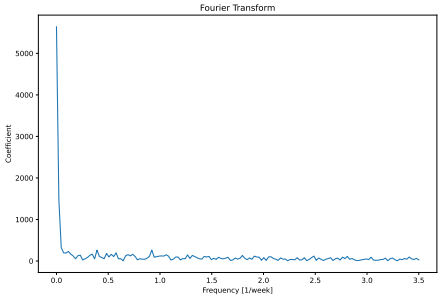


Figure 5.24: The coefficients given by the Fourier transform for the daily percentage of production.

Basic data is enhanced with the air temperature, the apparent temperature, the relative humidity,

the wind speed, the wind direction, the pressure altimeter, the visibility, the sky coverage, the diffuse horizontal irradiance, the direct normal irradiance, the global horizontal irradiance, the solar radiation, the UV index, the solar elevation angle, and the solar azimuth angle. To assess the relationship between these weather variables and the percentage of production, two correlation coefficients were used: Pearson's correlation coefficient and Spearman's rank correlation coefficient. Pearson's correlation coefficient measures the strength of the linear relationship between two variables, while Spearman's rank correlation coefficient measures the strength of the monotonic relationship. The pearsonr and spearmanr methods of the SciPy library were used to compute the correlation with weather data. The results showed that the hourly percentage of production had:

- a Pearson correlation coefficient of 0.5621 and a Spearman's rank correlation coefficient of 0.5185 with respect to the air temperature;
- a Pearson correlation coefficient of 0.5410 and a Spearman's rank correlation coefficient of 0.5032 with respect to the apparent temperature;
- a Pearson correlation coefficient of -0.6318 and a Spearman's rank correlation coefficient of -0.5969 with respect to the relative humidity;
- a Pearson correlation coefficient of 0.3105 and a Spearman's rank correlation coefficient of 0.3237 with respect to the wind speed;
- a Pearson correlation coefficient of -0.1919 and a Spearman's rank correlation coefficient of -0.2155 with respect to the wind direction;
- a Pearson correlation coefficient of -0.2092 and a Spearman's rank correlation coefficient of -0.2645 with respect to the pressure altimeter;
- a Pearson correlation coefficient of -0.0132 and a Spearman's rank correlation coefficient of 0.0355 with respect to the visibility;
- a Pearson correlation coefficient of -0.2822 and a Spearman's rank correlation coefficient of -0.2375 with respect to the sky coverage;
- a Pearson correlation coefficient of 0.8559 and a Spearman's rank correlation coefficient of 0.9340 with respect to the diffuse horizontal irradiance;
- a Pearson correlation coefficient of 0.8399 and a Spearman's rank correlation coefficient of 0.9315 with respect to the direct normal irradiance;
- a Pearson correlation coefficient of 0.8791 and a Spearman's rank correlation coefficient of 0.9357 with respect to the global horizontal irradiance;
- a Pearson correlation coefficient of 0.9073 and a Spearman's rank correlation coefficient of 0.9472 with respect to the solar radiation;
- a Pearson correlation coefficient of 0.8571 and a Spearman's rank correlation coefficient of 0.9392 with respect to the UV index;
- a Pearson correlation coefficient of 0.8107 and a Spearman's rank correlation coefficient of 0.9057 with respect to the solar elevation angle;
- a Pearson correlation coefficient of 0.0450 and a Spearman's rank correlation coefficient of 0.0465 with respect to the solar azimuth angle.

It can be noticed that both the coefficients indicate a moderate to strong correlation between the hourly percentage of production and several weather features, including the air temperature, the apparent temperature, the relative humidity, the diffuse horizontal irradiance, the direct normal irradiance, the global horizontal irradiance, the solar radiation, the UV index, and the solar elevation angle.

Therefore, these features are chosen to be incorporated into the prediction models, while the other weather features show weaker correlations and are not considered.

The results showed that the daily percentage of production had:

- a Pearson correlation coefficient of 0.7632 and a Spearman's rank correlation coefficient of 0.7721 with respect to the air temperature;
- a Pearson correlation coefficient of 0.7434 and a Spearman's rank correlation coefficient of 0.7471 with respect to the apparent temperature;
- a Pearson correlation coefficient of -0.7598 and a Spearman's rank correlation coefficient of -0.7474 with respect to the relative humidity;
- a Pearson correlation coefficient of 0.1773 and a Spearman's rank correlation coefficient of 0.3769 with respect to the wind speed;
- a Pearson correlation coefficient of -0.2405 and a Spearman's rank correlation coefficient of -0.2689 with respect to the wind direction;
- a Pearson correlation coefficient of -0.2506 and a Spearman's rank correlation coefficient of -0.2559 with respect to the pressure altimeter;
- a Pearson correlation coefficient of 0.2545 and a Spearman's rank correlation coefficient of 0.3259 with respect to the visibility;
- a Pearson correlation coefficient of -0.7004 and a Spearman's rank correlation coefficient of -0.6542 with respect to the sky coverage;
- a Pearson correlation coefficient of 0.6244 and a Spearman's rank correlation coefficient of 0.6939 with respect to the diffuse horizontal irradiance;
- a Pearson correlation coefficient of 0.6216 and a Spearman's rank correlation coefficient of 0.6923 with respect to the direct normal irradiance;
- a Pearson correlation coefficient of 0.6305 and a Spearman's rank correlation coefficient of 0.6988 with respect to the global horizontal irradiance;
- a Pearson correlation coefficient of 0.7878 and a Spearman's rank correlation coefficient of 0.8013 with respect to the solar radiation;
- a Pearson correlation coefficient of 0.8389 and a Spearman's rank correlation coefficient of 0.8680 with respect to the UV index;
- a Pearson correlation coefficient of 0.6502 and a Spearman's rank correlation coefficient of 0.7079 with respect to the solar elevation angle;
- a Pearson correlation coefficient of 0.1457 and a Spearman's rank correlation coefficient of 0.2813 with respect to the solar azimuth angle.

It can be noticed that both the coefficients decreased meaning that the mean weather data over the day is less correlated to the daily percentage of production, compared to the hourly granularity. Only the hourly granularity is considered since PV plants are highly correlated with weather data and the aggregation over the day loses part of this correlation.

If we were forecasting the production of each PV plant individually, the weather data at the specific locations of the plants would result in a higher correlation with the production data and likely allow the models to produce more accurate forecasts. However, since we are aggregating the production from multiple PV plants and looking for the overall percentage of production in the hour, using the weather data from a location located in the middle of the PV plants is also acceptable. In figure 5.25, the locations of the PV plants and of the weather station are reported.

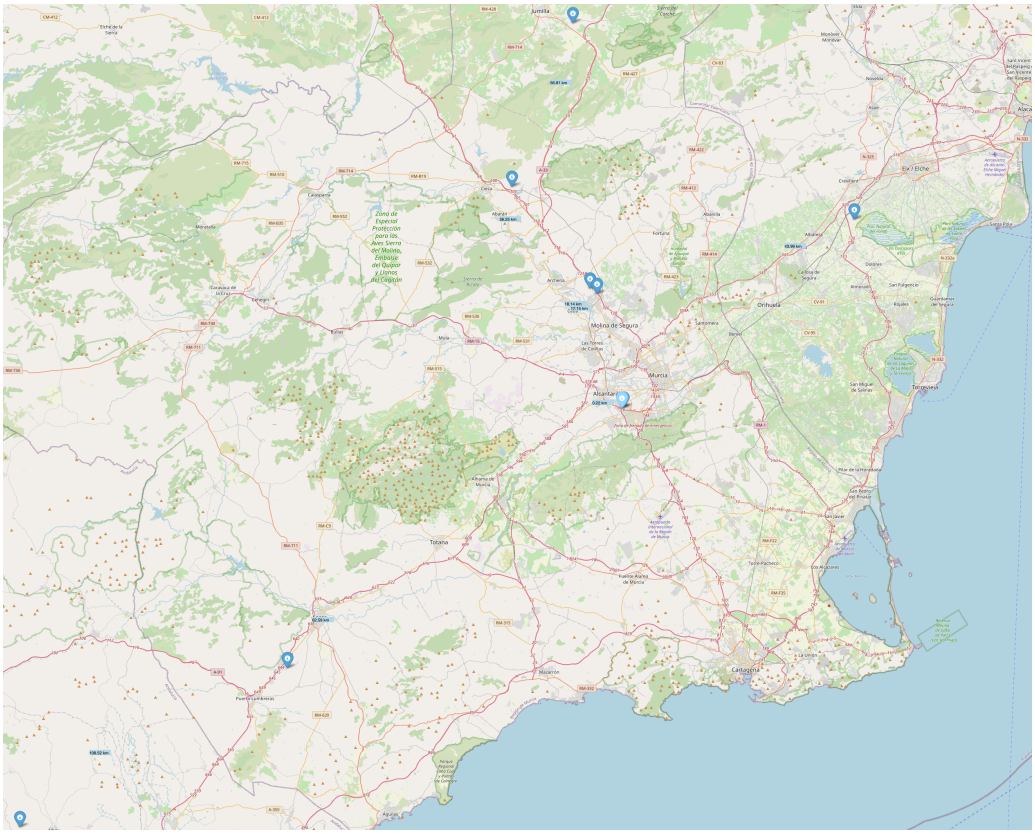


Figure 5.25: The map of the city of Murcia with indicated the location of the PV plants with a blue icon and of the revelation station with a lightblue icon. The distances between each PV plant and the weather station are reported in lightblue boxes near the PV plant locations.

After this data analysis, the specific parameters used in the models can be explained more in detail. The parameters for the different models were tested and verified on training data to result in the best-performing models, this was done by trying different values and configurations. The baseline approach is built considering the repetition of past days since data presents a high correlation with that time instant and could lead to a reasonable baseline performance to achieve. A simple ARIMA model considers the single hours without seasonally on the week period like for demand prediction since data does not present a relevant correlation with that time instants. The support vector regressor model uses a radial basis function kernel with a configuration of the C parameter (regularization parameter of squared l2 penalty) to 1.0 to penalize the complexity of the model, and the epsilon parameter defining the epsilon-tube for no penalty to 0.1 as a trade-off between accuracy and generalization. The hist gradient boosting regressor model uses as loss the absolute error since this was the supported metric closer to our evaluation metric. The learning rate was set to 0.1, 100 estimators were used and l2 regularization was not set. The parameters for tree definition are 31 as the maximum number of leaves, depth is not constrained, 20 as the minimum number of samples per leaf, and 255 as the maximum number of bins. For the extreme gradient boosting regressor model, tree-based models were used and the construction algorithm is automatically chosen by heuristic to choose the fastest method. It uses a squared loss since this was the supported metric closer to our evaluation metric. The learning rate was set to 0.3, 100 estimators were used and l2 regularization was set with a weight of 1.0. The parameters for tree definition are 6 as the maximum depth, and the maximum number of leaves is not constrained. The Prophet model was built keeping the default parameters for automatically detecting seasonalities and best fitting the training data.

The LSTM model was designed as a 2-layer model with 32 Bidirectional LSTM units in the first layer which use ReLU activation function and sigmoid as recurrent activation function with both dropout and recurrent dropout of 0.02. The second layer is composed of 16 Bidirectional LSTM units which use ReLU activation function and sigmoid as recurrent activation function without dropout and recurrent dropout. The output of the layer goes inside a dense unit to output the final prediction.

The model is trained using mean absolute error as loss and Nadam as optimizer with a learning rate of 0.005.

The GRU model was designed as a 2-layer model with 32 Bidirectional GRU units in the first layer which use ReLU activation function and sigmoid as recurrent activation function with both dropout and recurrent dropout of 0.02. Subsequently, the second layer is composed of 16 Bidirectional GRU units which use ReLU activation function and sigmoid as recurrent activation function without dropout and recurrent dropout. The output of the layer goes inside a dense unit to output the final prediction. The model is trained using mean absolute error as loss and Nadam as optimizer with a learning rate of 0.005.

The CNN model was designed as a 2-layer model with a first layer composed of 24 1D Convolutional units with a kernel size of 5 which uses ReLU activation function. A 1D max polling operation is applied before entering the second layer composed of 16 1D Convolutional units with a kernel size of 3 which use ReLU activation function. A 1D max polling operation is applied before the flatting operation and entering the final dense unit to output the final prediction. The model is trained using mean absolute error as loss and Nadam as optimizer with a learning rate of 0.005.

The TFT model was configured with 2 LSTM layers with 16 as hidden size, 4 as attention size, and 8 as hidden continuous size. The dropout is set to 0.02. The model is trained using mean absolute error as loss and Nadam as optimizer with a learning rate of 0.005.

12 splits were used for block validation with a test size of a week each time, namely 7 days of data. The models start with 5280 entries as training and predict every time 168 entries until reaching the last prediction instant where the model has a training size of 7128. The results for hourly aggregation are reported in table 5.6.

Model	**Blocked k-fold cross-validation MAE percentage**	**Test on the last split MAE percentage**
GRU	5.70 ± 1.78	2.79
HistGradientBoostingRegressor	4.16 ± 0.99	2.98
SVR	5.20 ± 1.12	3.12
CNN	5.98 ± 1.52	3.27
XGBRegressor	4.57 ± 0.97	3.42
LSTM	5.23 ± 1.46	3.51
TFT	4.70 ± 1.39	3.60
ARIMA	5.09 ± 1.35	3.64
Prophet	7.90 ± 1.62	4.38
One Day Baseline	6.81 ± 3.14	4.58

Table 5.6: Table summarizing the results for hourly aggregation.

The results for hourly aggregation at forecasting time on the last time slot including also the combinations of different techniques and the AutoML approach are reported in table 5.7. The combination of the previous techniques was performed by combining one-day baseline and prophet with possibly one of LSTM, GRU, or CNN with the intent of trying to improve the robustness of DL models with some stable baselines and prophet model which internally automatically deals with seasonalities. The AutoML approach uses the mean MAPE forecasting as loss and it is launched for 10 hours with a maximum function evaluation time of 2 hours. Also, the mean MAE forecasting was used but provided worse results. Then the best ensemble of the found models is returned and can be used to forecast.

Model	Forecast on the last split MAE percentage
GRU	2.79
HistGradientBoostingRegressor	2.86
SVR	3.12
CNN	3.27
XGBRegressor	3.35
LSTM + One Day Baseline	3.37
GRU + One Day Baseline	3.45
CNN + One Day Baseline	3.48
LSTM	3.51
TFT	3.60
LSTM + One Day Baseline + Prophet	3.60
ARIMA	3.64
CNN + One Day Baseline + Prophet	3.67
GRU + One Day Baseline + Prophet	3.70
One Day Baseline + Prophet	4.38
Prophet	4.38
One Day Baseline	4.58
AutoML	5.89

Table 5.7: Table summarizing the results for hourly aggregation at forecasting time.

Analyze the forecasts of the models for the electricity production forecasting task (predictive analytics) ...

The forecasts produced by the GRU model for the last split in comparison to the actual values are reported in figure 5.26. It can be noticed that ... The absolute error of forecasts produced by the GRU model for the last split in comparison to the actual values is reported in figure 5.27. It can be shown that ... Finally, the scatterplot representing the error of the forecasts produced by the GRU model for the last split in comparison to the actual values is reported in figure 5.28. This scatterplot demonstrates that ...

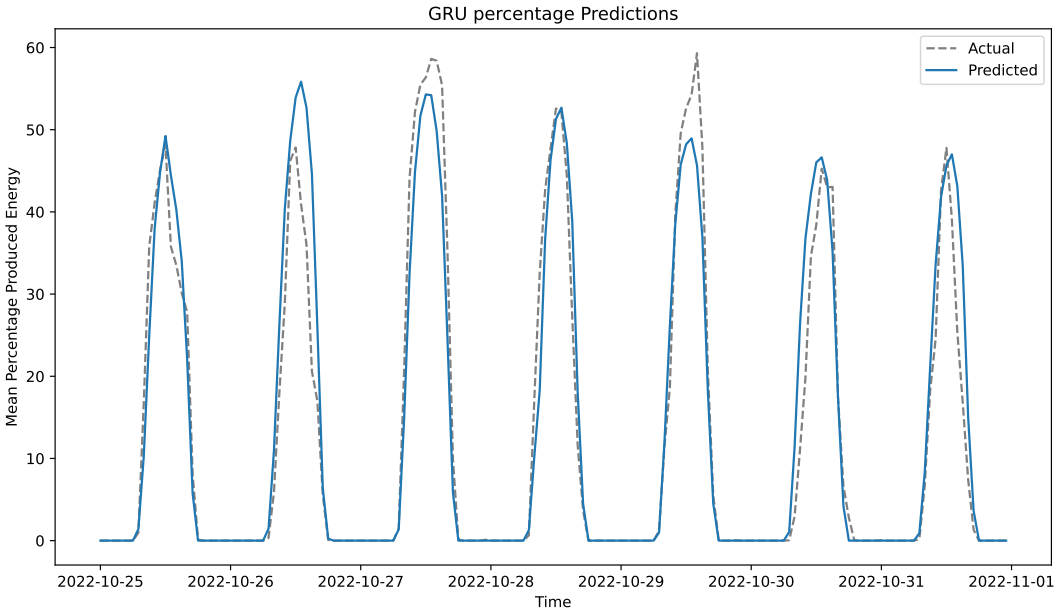


Figure 5.26: The forecasts produced by the gated recurrent unit model for the last split in comparison to the actual values.

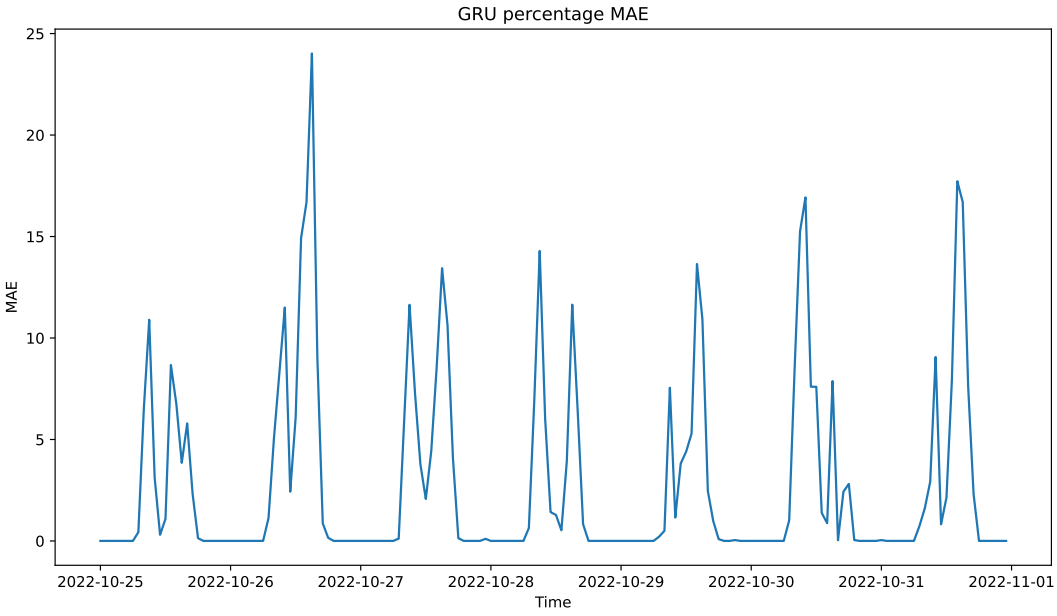


Figure 5.27: The absolute error of forecasts produced by the gated recurrent unit model for the last split in comparison to the actual values.

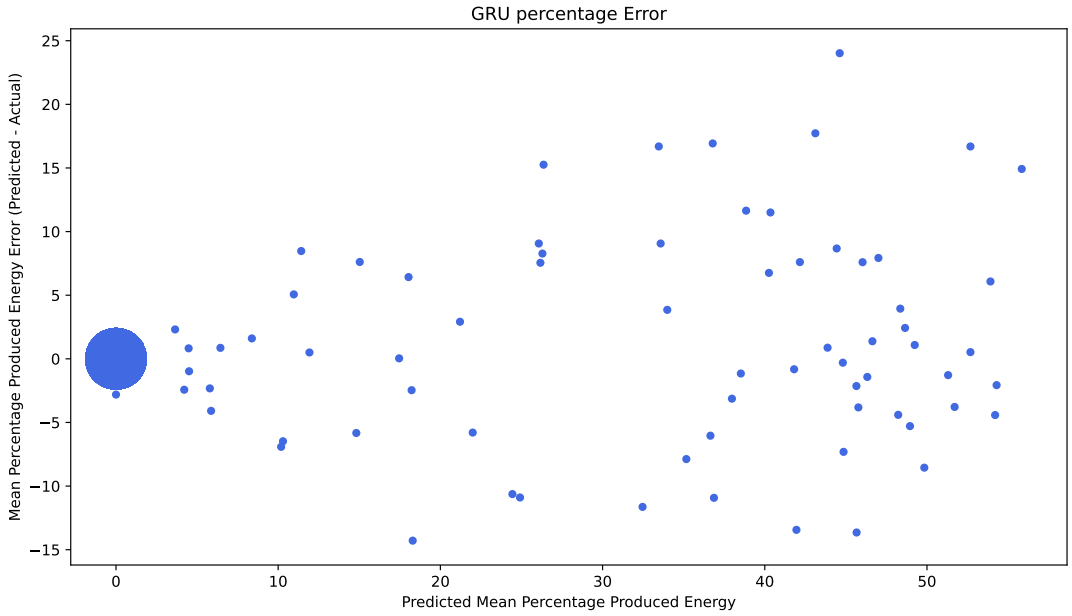
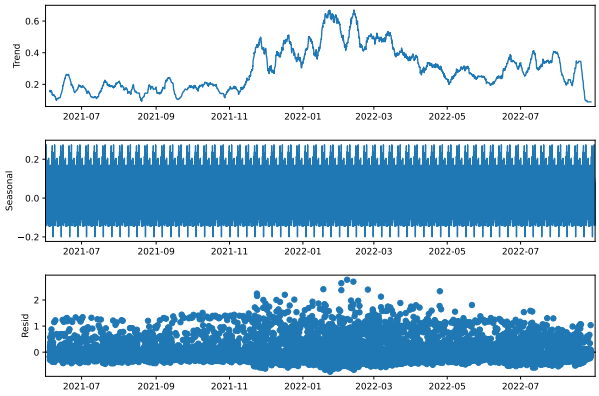


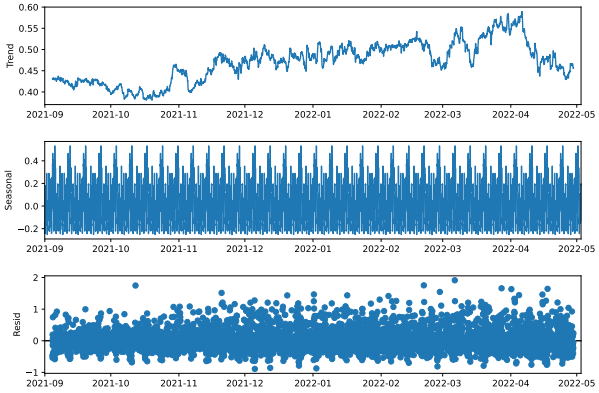
Figure 5.28: The scatterplot representing the error of the forecasts produced by the gated recurrent unit model for the last split in comparison to the actual values.

5.5 Consumption baseline forecasting

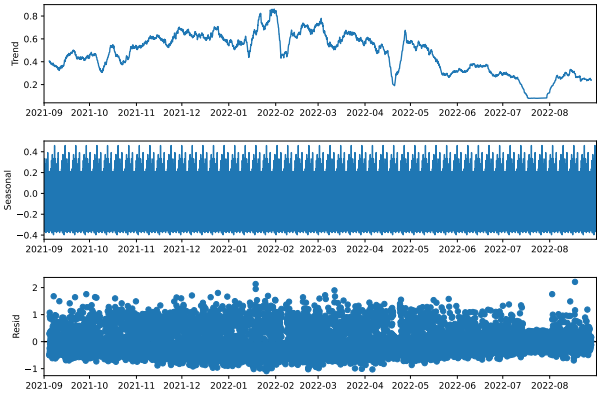
The consumption data of the three customers are analyzed to get some descriptive analytics before finding adequate models to forecast the consumption baseline of each customer. The time series decompositions using an additive model of the hourly consumption of the three customers considering as period of the time series a week are reported in figure 5.29a, figure 5.29b, and figure 5.29c. All three decompositions show a considerable amount of noise, which is dominant in magnitude compared to the trend and seasonality. For all the customers the seasonal component appears to not have a significant impact on the time series, with the exception of the second customer for which it is twice high in magnitude with respect to the others. The trend component has a slight impact, for the first customer it is possible to see a peak in the winter, for the second customer it is present a peak in April, and for the third customer there is not a clear trend.



(a) First customer.



(b) Second customer.

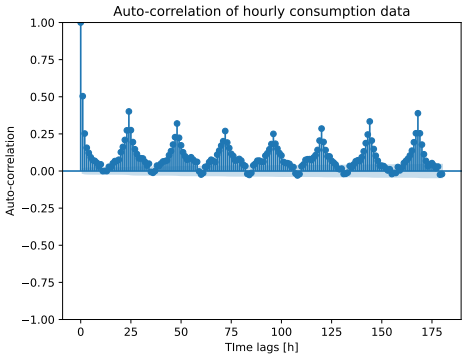


(c) Third customer.

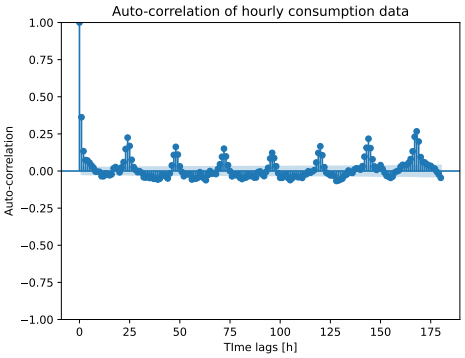
Figure 5.29: The time series decompositions of the hourly consumption of the three customers considering as period of the time series a week.

The auto-correlations of the hourly consumption of the three customers are reported in figure 5.30a, figure 5.30b, and figure 5.30c. All the customers' consumptions are not very auto-correlated, it can be shown that the maximum auto-correlation value is around 0.5 in the closest time lag and that there is a peak every 24 hours, with a slightly greater value at one week distance. This indicates that the consumption data from the closest time lag, as well as those corresponding to the same hour in the preceding days and even better in the preceding weeks, may be valuable features for predicting a time instant's demand. A reasonable balance can be achieved by incorporating the consumption data from the past 14 days.

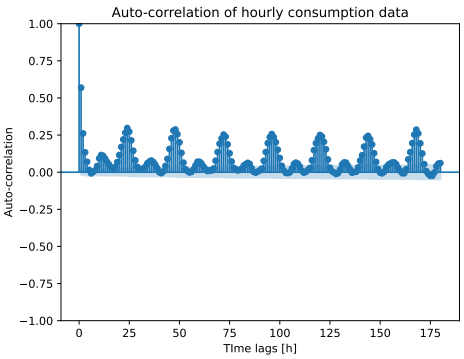
The coefficients given by the Fourier transform for the hourly consumption of the three customers are reported in figure 5.31a, figure 5.31b, and figure 5.31c. The graphical representations show for all the customers a main frequency at the daily periodicity. Other minor peaks are present mostly at multiples of the 1/day frequency, in particular for the third customer where the 2/day frequency is almost equal to the 1/day.



(a) First customer.

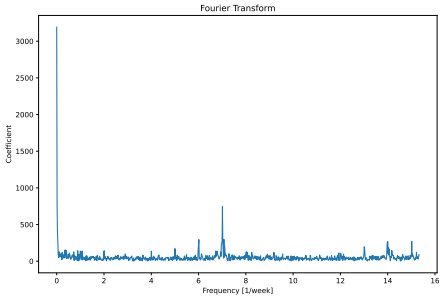


(b) Second customer.

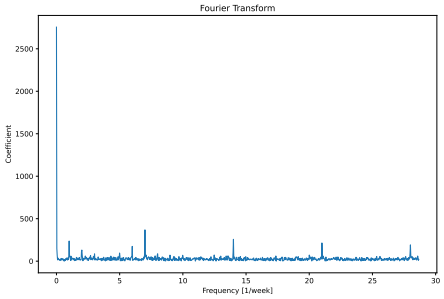


(c) Third customer.

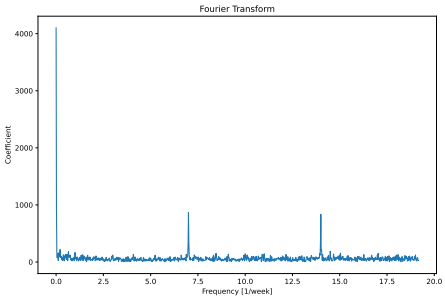
Figure 5.30: The auto-correlations of the hourly consumption of the three customers.



(a) First customer.



(b) Second customer.



(c) Third customer.

Figure 5.31: The coefficients given by the Fourier transform of the hourly consumption of the three customers.

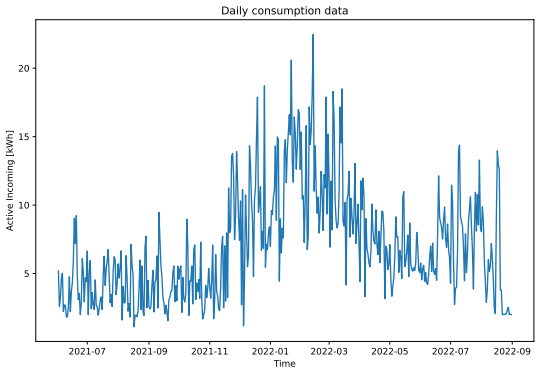
The daily consumption of the three customers are reported in figure 5.32a, figure 5.32b, and

figure 5.32c.

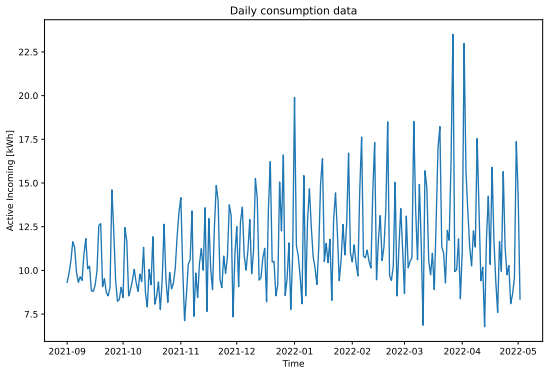
The time series decompositions using an additive model of the daily consumption of the three customers considering as period of the time series a week are reported in figure 5.33a, figure 5.33b, and figure 5.33c. The decomposition is consistent with the hourly aggregated consumption over the customers.

The auto-correlations of the daily consumption of the three customers are reported in figure 5.34a, figure 5.34b, and figure 5.34c. With the daily aggregation, all the customers' consumptions show are slightly high value with respect to the hourly granularity. It can be shown that for the first and third customers, the auto-correlation value is slightly high in the closest time lag and then decreases with the days, with a slightly greater value at one week distance. Instead, for the second customer the only relevant values are only at a multiple of 7 days.

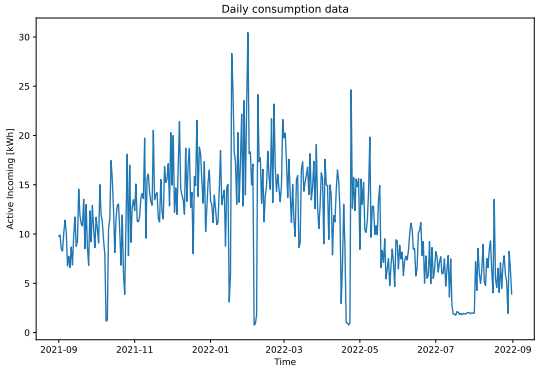
The coefficients given by the Fourier transform for the daily consumption of the three customers are reported in figure 5.35a, figure 5.35b, and figure 5.35c. The graphical representation exhibits that there are no main frequencies in the daily consumption of the first and third customers. Instead, for the second customer there is a small peak at the daily periodicity.



(a) First customer.



(b) Second customer.



(c) Third customer.

Figure 5.32: The daily consumption of the three customers.

As can be noticed from the data, there is high variability in consumption and low auto-correlation, this suggests how it is difficult to produce highly accurate results on a single customer level. With just the time series of a few users, it is very difficult to learn a well-performing model, having more users it could be possible to learn certain generic trends or standard behaviors.

Basic data is enhanced with the air temperature, the apparent temperature, and the relative humidity since they are considered the only weather features capable of influencing customers' energy consumption. To assess the relationship between these weather variables and the consumption of the three customers, two correlation coefficients were used: Pearson's correlation coefficient and Spearman's rank correlation coefficient. Pearson's correlation coefficient measures the strength of the linear

relationship between two variables, while Spearman's rank correlation coefficient measures the strength of the monotonic relationship. The pearsonr and spearmanr methods of the SciPy library were used to compute the correlation with weather data. The results showed that the hourly consumption of the three customers had:

- a Pearson correlation coefficient of -0.2491, 0.0240, and -0.1338 respectively and a Spearman's rank correlation coefficient of -0.2088, 0.1304, and -0.0196 respectively with respect to the air temperature;

- a Pearson correlation coefficient of -0.2434, 0.0266, and -0.1356 respectively and a Spearman's rank correlation coefficient of -0.2082, 0.1305, and -0.0207 respectively with respect to the apparent temperature;

- a Pearson correlation coefficient of 0.1193, -0.0845, and -0.0985 respectively and a Spearman's rank correlation coefficient of 0.1491, -0.1048, and -0.1123 respectively with respect to the relative humidity.

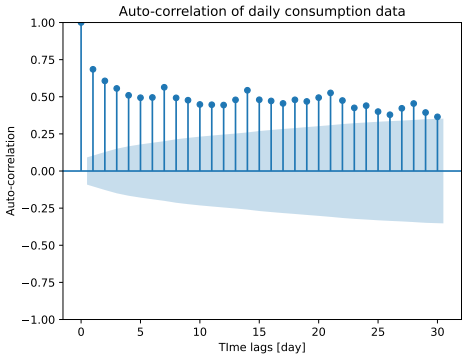
It can be noticed that both the coefficients indicate a weak correlation between the weather variables and the consumption of the three customers, nevertheless, it may still be useful to incorporate weather data into the prediction models since also the auto-correlation values are not particularly high.

(a) First customer.

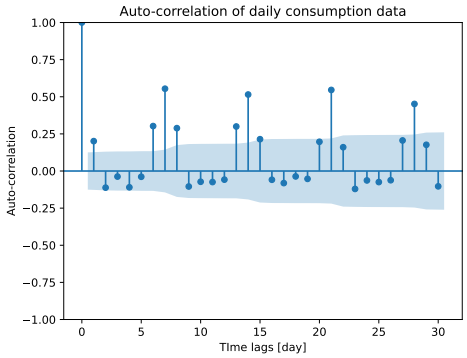
(b) Second customer.

(c) Third customer.

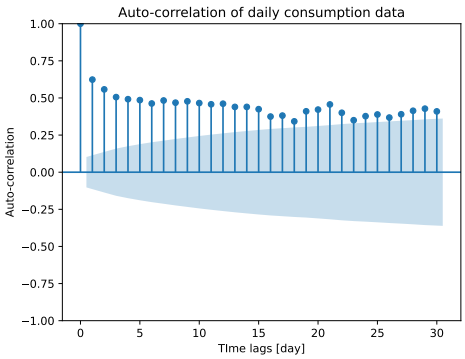
Figure 5.33: The time series decompositions of the daily consumption of the three customers considering as period of the time series a week.



(a) First customer.

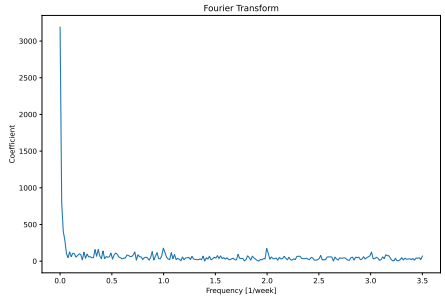


(b) Second customer.

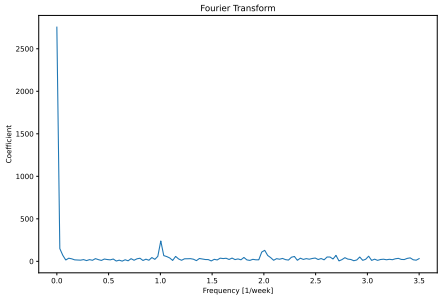


(c) Third customer.

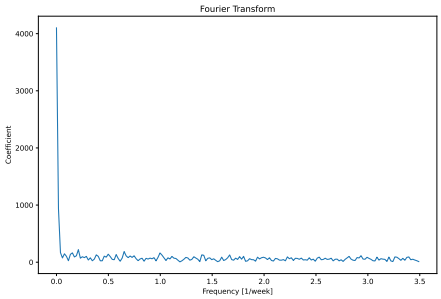
Figure 5.34: The auto-correlations of the daily consumption of the three customers.



(a) First customer.



(b) Second customer.



(c) Third customer.

Figure 5.35: The coefficients given by the Fourier transform of the daily consumption of the three customers.

The results showed that the daily consumption of the three customers had:

- a Pearson correlation coefficient of -0.5542, -0.2341, and -0.6727 respectively and a Spearman's rank correlation coefficient of -0.5164, -0.2592, and -0.6978 respectively with respect to the air temperature;

- a Pearson correlation coefficient of -0.5361, -0.2329, and -0.6787 respectively and a Spearman's rank correlation coefficient of -0.5166, -0.2584, and -0.7004 respectively with respect to the apparent temperature;

- a Pearson correlation coefficient of 0.1264, 0.0199, and 0.1407 respectively and a Spearman's rank correlation coefficient of 0.1412, 0.0506, and 0.2104 respectively with respect to the relative humidity.

It can be noticed that both the coefficients increased considering the daily data meaning that the mean weather data over the day is more correlated to the daily consumption of the three considered customers, compared to the hourly granularity.

As described in chapter 4, tariff aggregation has also been thought of as a possible aggregation. Tariff aggregation assumes that the tariff is a control signal capable of modifying consumption. However, after some initial experimental results with this aggregation showing unsatisfactory performance, this aggregation was not further investigated. Though, this aggregation can be investigated in a possible future work to understand whether it could be applicable to produce good results.

After this data analysis, the specific parameters used in the models can be explained more in detail. The parameters for the different models were tested and verified on training data to result in the best-performing models, this was done by trying different values and configurations. The baseline approaches are built considering the repetition of past days and weeks since data presents a high correlation with that time instants and could lead to a reasonable baseline performance to achieve. The SARIMA model considers the week as the period for seasonal differencing since data presents a high correlation with that time instants and the model can try to take advantage of the weekly seasonality. The support vector regressor model uses a radial basis function kernel with a configuration of the C parameter (regularization parameter of squared l2 penalty) to 1.0 to penalize the complexity of the model, and the epsilon parameter defining the epsilon-tube for no penalty to 0.1 as a trade-off between accuracy and generalization. The hist gradient boosting regressor model uses as loss the absolute error since this was the supported metric closer to our evaluation metric. The learning rate was set to 0.1, 100 estimators were used and l2 regularization was not set. The parameters for tree definition are 31 as the maximum number of leaves, depth is not constrained, 20 as the minimum number of samples per leaf, and 255 as the maximum number of bins. For the extreme gradient boosting regressor model, tree-based models were used and the construction algorithm is automatically chosen by heuristic to choose the fastest method. It uses a squared loss since this was the supported metric closer to our evaluation metric. The learning rate was set to 0.3, 100 estimators were used and l2 regularization was set with a weight of 1.0. The parameters for tree definition are 6 as the maximum depth, and the maximum number of leaves is not constrained. The Prophet model was built keeping the default parameters for automatically detecting seasonalities and best fitting the training data.

The LSTM model was designed as a 2-layer model with 32 Bidirectional LSTM units in the first layer which use ReLU activation function and sigmoid as recurrent activation function with both dropout and recurrent dropout of 0.02. The second layer is composed of 16 Bidirectional LSTM units which use ReLU activation function and sigmoid as recurrent activation function without dropout and recurrent dropout. The output of the layer goes inside a dense unit to output the final prediction. The model is trained using a combination of mean absolute error and mean absolute percentage error as loss and Nadam as optimizer with a learning rate of 0.005.

The GRU model was designed as a 2-layer model with 32 Bidirectional GRU units in the first layer which use ReLU activation function and sigmoid as recurrent activation function with both dropout and recurrent dropout of 0.02. Subsequently, the second layer is composed of 16 Bidirectional GRU units which use ReLU activation function and sigmoid as recurrent activation function without dropout and recurrent dropout. The output of the layer goes inside a dense unit to output the final prediction.

The model is trained using a combination of mean absolute error and mean absolute percentage error as loss and Nadam as optimizer with a learning rate of 0.005.

The CNN model was designed as a 2-layer model with a first layer composed of 32 1D Convolutional units with a kernel size of 5 which uses ReLU activation function. A 1D max polling operation is applied before entering the second layer composed of 16 1D Convolutional units with a kernel size of 3 which use ReLU activation function. A 1D max polling operation is applied before the flatting operation and entering the final dense unit to output the final prediction. The model is trained using a combination of mean absolute error and mean absolute percentage error as loss and Nadam as optimizer with a learning rate of 0.005.

The TFT model was configured with 2 LSTM layers with 16 as hidden size, 4 as attention size, and 8 as hidden continuous size. The dropout is set to 0.02. The model is trained using mean absolute percentage error as loss and Nadam as optimizer with a learning rate of 0.005.

12 splits were used for block validation with a test size of a week each time, namely 7 days of data. For the hourly granularity, the models start with the total number of entries for the customer minus 2016 entries as training and predict every time 168 entries until reaching the last prediction instant where the model has a training size of the total number of entries for the customer minus the last 168 entries. The results for hourly aggregation for the second customer are reported in table 5.8. Only this customer is reported since it presents the best results and the models were optimized for this customer.

Model	**Blocked k-fold cross-validation MAPE — MAE [kWh]**	**Test on the last split MAPE — MAE [kWh]**
TFT	47.33 ± 5.57 — 0.287 ± 0.028	44.19 — 0.231
HistGradientBoostingRegressor	53.95 ± 7.02 — 0.260 ± 0.023	50.41 — 0.231
SVR	60.30 ± 6.81 — 0.278 ± 0.032	60.41 — 0.243
SARIMA	74.17 ± 10.01 — 0.271 ± 0.021	78.02 — 0.244
XGBRegressor	73.71 ± 9.54 — 0.275 ± 0.024	68.07 — 0.245
Prophet	83.84 ± 10.23 — 0.280 ± 0.020	81.92 — 0.249
GRU	58.88 ± 19.55 — 0.309 ± 0.041	42.55 — 0.257
LSTM	111.87 ± 212.07 — 0.464 ± 0.516	44.23 — 0.266
4 Week Baseline	74.06 ± 12.51 — 0.275 ± 0.027	79.07 — 0.266
12 Week Baseline	74.30 ± 13.66 — 0.266 ± 0.024	89.23 — 0.278
One Week Baseline	82.69 ± 12.83 — 0.326 ± 0.030	70.48 — 0.281
One Day Baseline	75.80 ± 13.96 — 0.318 ± 0.035	71.25 — 0.292
CNN	148.16 ± 30.05 — 0.510 ± 0.105	70.19 — 0.318

Table 5.8: Table summarizing the results for hourly aggregation.

The AutoML approach uses the mean MAPE forecasting as loss and it is launched for 10 hours with a maximum function evaluation time of 2 hours. Then the best ensemble of the found models is returned and can be used to forecast. The AutoML approach obtained a one-week MAPE of 61.64 and a MAE of 0.338 [kWh] after a total run of 10 hours.

Analyze the forecasts of the models for the electricity demand forecasting task (predictive analytics) ...

The forecasts produced by the TFT model for the last split in comparison to the actual values are reported in figure 5.36. It can be noticed that ... The absolute error of forecasts produced by the TFT model for the last split in comparison to the actual values is reported in figure 5.37. While the absolute percentage error of forecasts produced by the TFT model for the last split in comparison to the actual values is reported in figure 5.38. It can be shown that ... Finally, the scatterplot representing the error of the forecasts produced by the TFT model for the last split in comparison to the actual values is reported in figure 5.39. This scatterplot demonstrates that ...

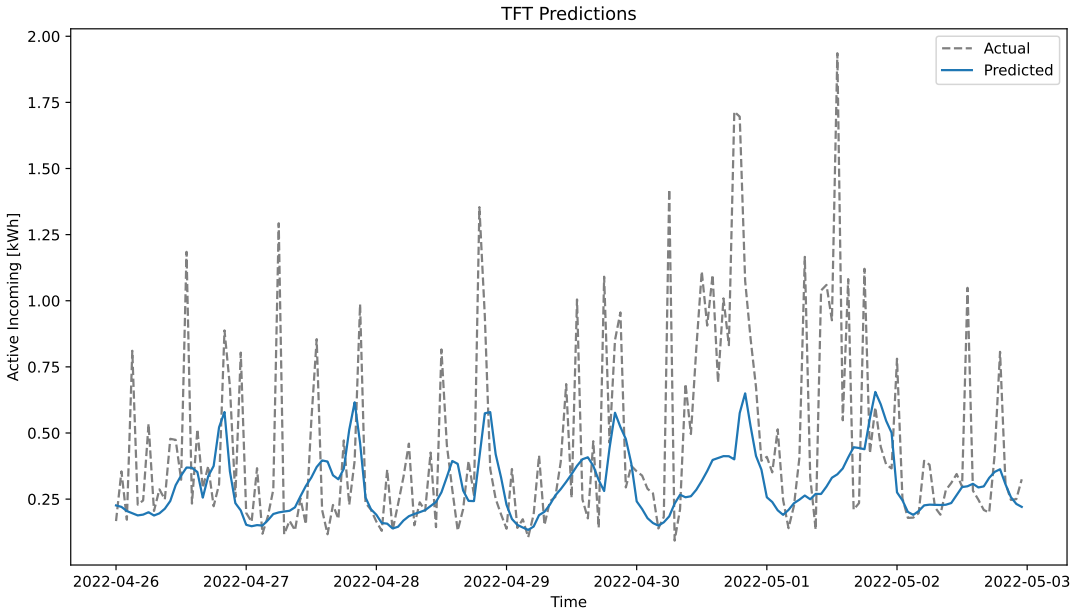


Figure 5.36: The forecasts produced by the temporal fusion transoformer model for the last split in comparison to the actual values.

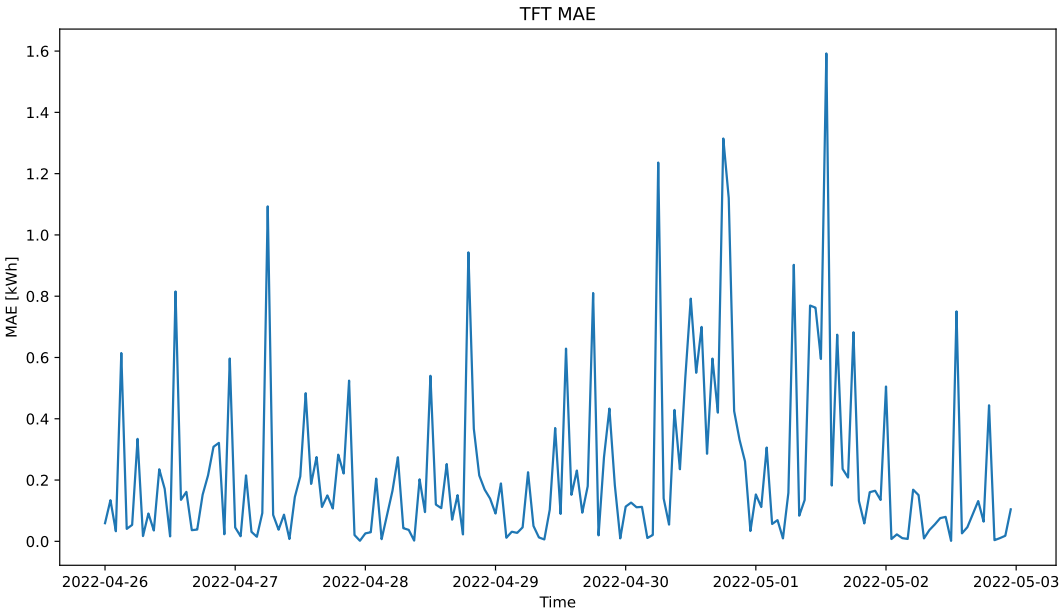


Figure 5.37: The absolute error of forecasts produced by the temporal fusion transoformer model for the last split in comparison to the actual values.

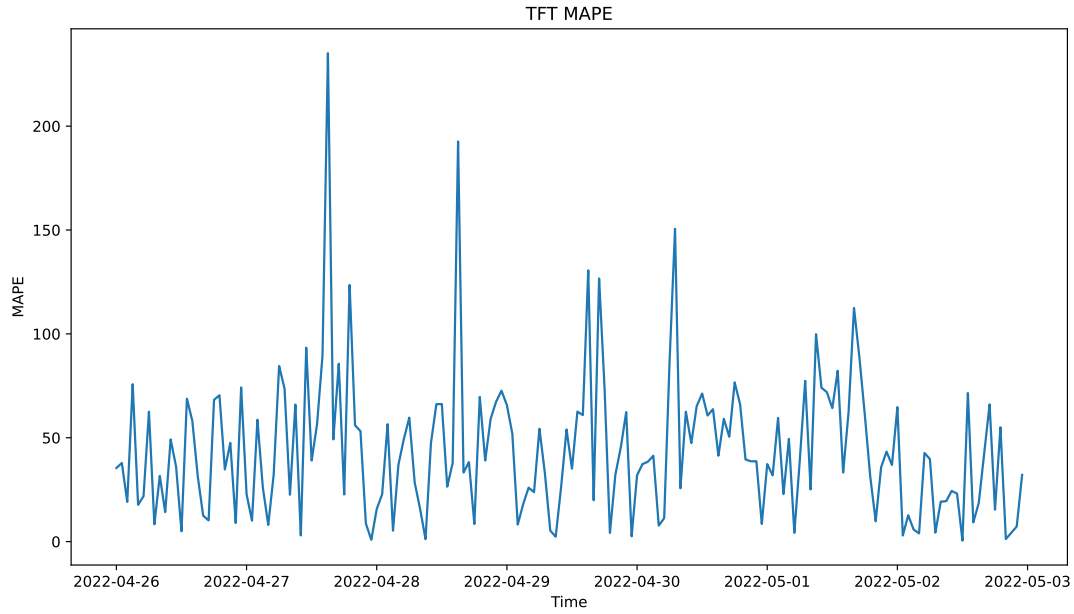


Figure 5.38: The absolute percentage error of forecasts produced by the temporal fusion transoformer model for the last split in comparison to the actual values.

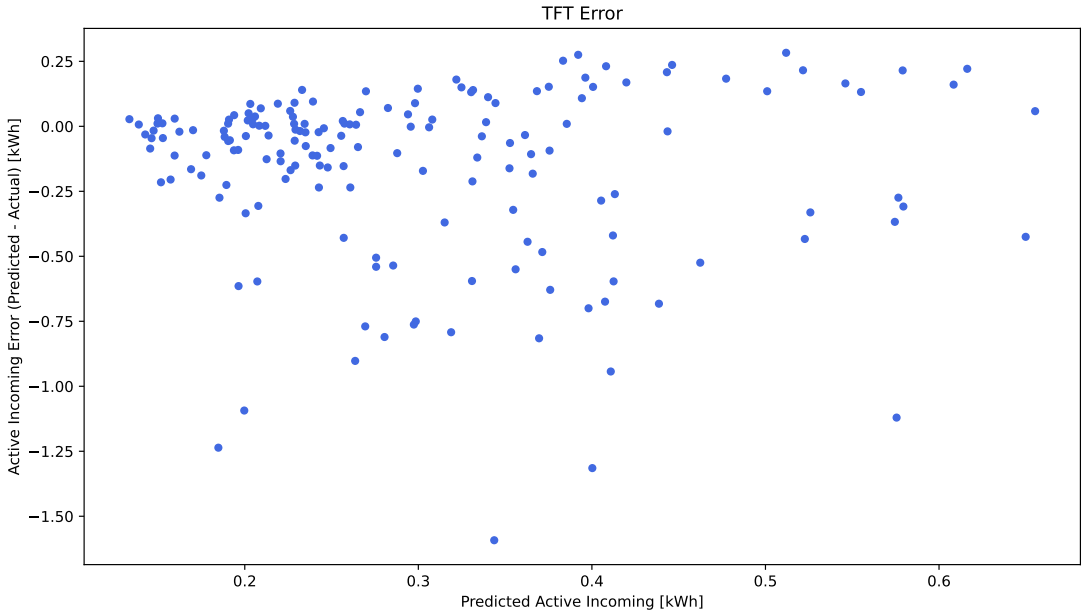


Figure 5.39: The scatterplot representing the error of the forecasts produced by the temporal fusion transoformer model for the last split in comparison to the actual values.

For the daily granularity, the models start with the total number of entries for the customer minus 84 entries as training and predict every time 30 entries until reaching the last prediction instant where the model has a training size of the total number of entries for the customer minus the last 7 entries. The results for daily aggregation for the second customer are reported in table 5.9.

Model	Blocked k-fold cross-validation MAPE — MAE [kWh]	Test on the last split MAPE — MAE [kWh]
4 Week Baseline	16.16 ± 4.88 — 1.906 ± 0.572	14.40 — 1.318
One Week Baseline	19.56 ± 7.45 — 2.343 ± 0.899	14.45 — 1.554
CNN	28.45 ± 17.47 — 3.320 ± 1.939	13.46 — 1.751
SARIMA	16.38 ± 4.82 — 1.896 ± 0.454	17.07 — 1.773
TFT	16.67 ± 6.33 — 2.002 ± 0.643	14.55 — 1.786
HistGradientBoostingRegressor	15.36 ± 3.72 — 1.889 ± 0.355	16.80 — 1.877
12 Week Baseline	15.56 ± 4.43 — 1.861 ± 0.460	19.99 — 2.006
XGBRegressor	17.62 ± 4.88 — 2.135 ± 0.558	21.38 — 2.111
One Day Baseline	20.36 ± 3.93 — 2.657 ± 0.551	18.95 — 2.432
Prophet	17.36 ± 5.18 — 1.947 ± 0.426	25.45 — 2.458
GRU	18.68 ± 3.36 — 2.463 ± 0.534	22.83 — 2.643
SVR	18.10 ± 3.24 — 2.449 ± 0.552	23.44 — 2.694
LSTM	18.96 ± 3.52 — 2.500 ± 0.616	28.83 — 3.060

Table 5.9: Table summarizing the results for daily aggregation.

Instead, the AutoML approach obtained a one-week MAPE of 19.88 and a MAE of 2.197 [kWh] after a total run of 10 hours.

The forecasts produced by the four-week baseline for the last split in comparison to the actual values are reported in figure 5.40. It can be noticed that ... The absolute error of forecasts produced by the four-week baseline for the last split in comparison to the actual values is reported in figure 5.41. While the absolute percentage error of forecasts produced by the four-week baseline for the last split in comparison to the actual values is reported in figure 5.42. It can be shown that ... Finally, the scatterplot representing the error of the forecasts produced by the four-week baseline for the last split in comparison to the actual values is reported in figure 5.43. This scatterplot demonstrates that ...

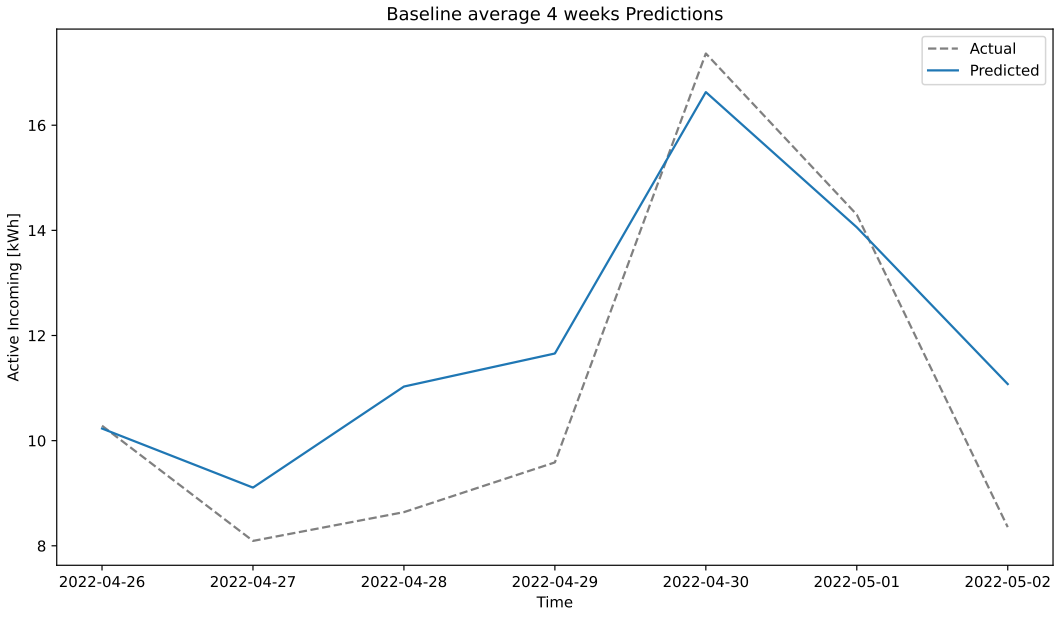


Figure 5.40: The forecasts produced by the four-week baseline for the last split in comparison to the actual values.

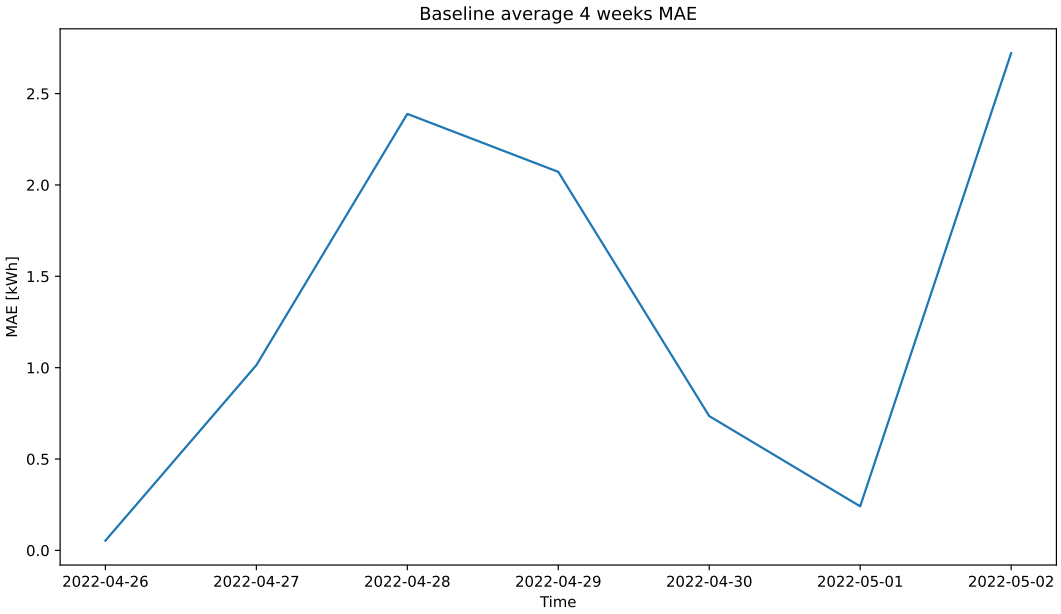


Figure 5.41: The absolute error of forecasts produced by the four-week baseline for the last split in comparison to the actual values.

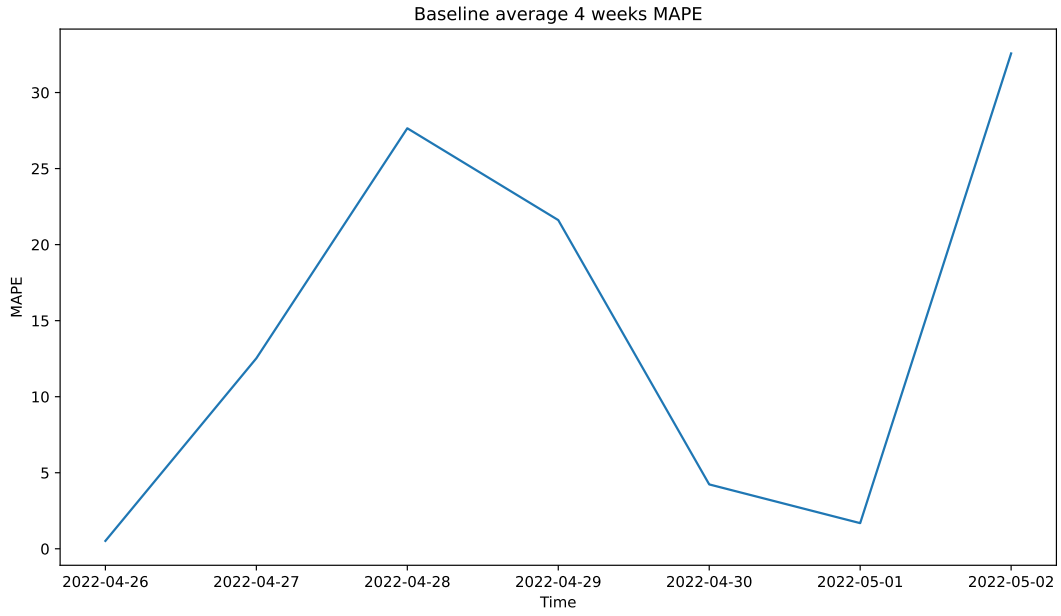


Figure 5.42: The absolute percentage error of forecasts produced by the four-week baseline for the last split in comparison to the actual values.

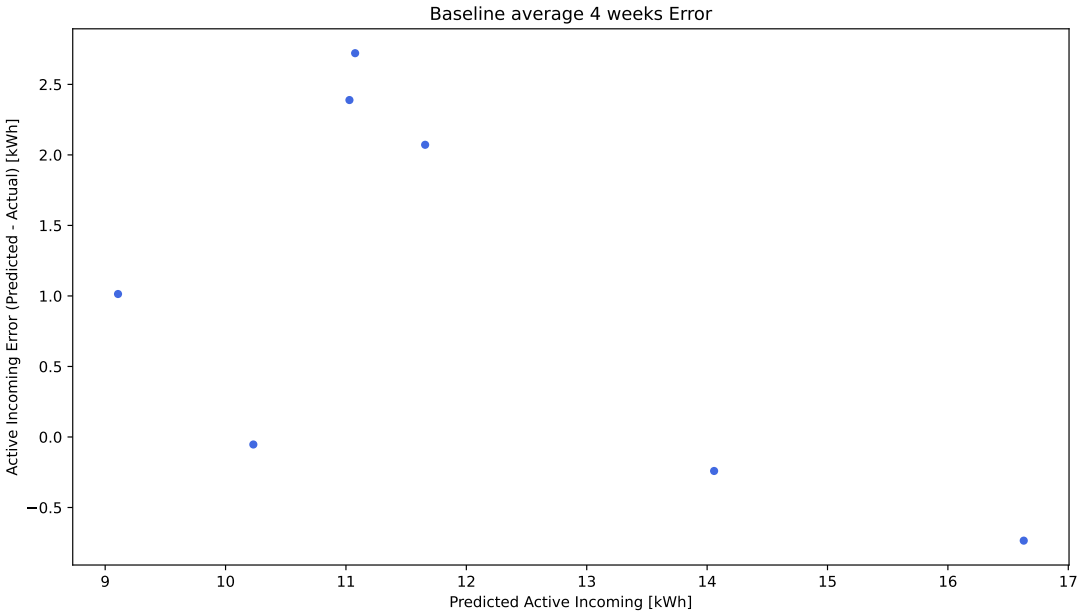


Figure 5.43: The scatterplot representing the error of the forecasts produced by the four-week baseline for the last split in comparison to the actual values.

6 Conclusions

This chapter reports the conclusions and summary of the work done. At the end of the thesis, some ideas for future works are suggested.

6.1 Summary

Summary of the work done ...

6.2 Future works

Ideas for future works ...

Bibliography

[1] Muhammad Waseem Ahmad, Monjur Mourshed, and Yacine Rezgui. "Tree-based ensemble methods for predicting PV power generation and their comparison with support vector regression". In: *Energy* 164 (2018), pp. 465–474. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2018.08.207. URL: https://www.sciencedirect.com/science/article/pii/S0360544218317432.

[2] Tanveer Ahmad, Huanxin Chen, Yabin Guo, and Jiangyu Wang. "A comprehensive overview on the data driven and large scale based approaches for forecasting of building energy demand: A review". In: *Energy and Buildings* 165 (2018), pp. 301–320. ISSN: 0378-7788. DOI: https://doi.org/10.1016/j.enbuild.2018.01.017. URL: https://www.sciencedirect.com/science/article/pii/S0378778817329225.

[3] Nesreen K. Ahmed, Amir F. Atiya, Neamat El Gayar, and Hisham El-Shishiny. "An Empirical Comparison of Machine Learning Models for Time Series Forecasting". In: *Econometric Reviews* 29.5-6 (2010), pp. 594–621. DOI: 10.1080/07474938.2010.481556. URL: https://doi.org/10.1080/07474938.2010.481556.

[4] R. Ahmed, V. Sreeram, Y. Mishra, and M.D. Arif. "A review and evaluation of the state-of-the-art in PV solar power forecasting: Techniques and optimization". In: *Renewable and Sustainable Energy Reviews* 124 (2020), p. 109792. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2020.109792. URL: https://www.sciencedirect.com/science/article/pii/S1364032120300885.

[5] Mohanad S. Al-Musaylh, Ravinesh C. Deo, Jan F. Adamowski, and Yan Li. "Short-term electricity demand forecasting with MARS, SVR and ARIMA models using aggregated demand data in Queensland, Australia". In: *Advanced Engineering Informatics* 35 (2018), pp. 1–16. ISSN: 1474-0346. DOI: https://doi.org/10.1016/j.aei.2017.11.002. URL: https://www.sciencedirect.com/science/article/pii/S1474034617301477.

[6] Mohammad H. Alobaidi, Fateh Chebana, and Mohamed A. Meguid. "Robust ensemble learning framework for day-ahead forecasting of household based energy consumption". In: *Applied Energy* 212 (2018), pp. 997–1012. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2017.12.054. URL: https://www.sciencedirect.com/science/article/pii/S0306261917317695.

[7] Ahmad Alsharef, Sonia, Karan Kumar, and Celestine Iwendi. "Time Series Data Modeling Using Advanced Machine Learning and AutoML". In: *Sustainability* 14.22 (2022). ISSN: 2071-1050. DOI: 10.3390/su142215292. URL: https://www.mdpi.com/2071-1050/14/22/15292.

[8] K.P. Amber, R. Ahmad, M.W. Aslam, A. Kousar, M. Usman, and M.S. Khan. "Intelligent techniques for forecasting electricity consumption of buildings". In: *Energy* 157 (2018), pp. 886–893. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2018.05.155. URL: https://www.sciencedirect.com/science/article/pii/S036054421830999X.

[9] J. Antonanzas, N. Osorio, R. Escobar, R. Urraca, F.J. Martinez de Pison, and F. Antonanzas-Torres. "Review of photovoltaic power forecasting". In: *Solar Energy* 136 (2016), pp. 78–111. ISSN: 0038-092X. DOI: https://doi.org/10.1016/j.solener.2016.06.069. URL: https://www.sciencedirect.com/science/article/pii/S0038092X1630250X.

[10] J.S. Armstrong. *Principles of Forecasting: A Handbook for Researchers and Practitioners*. International Series in Operations Research & Management Science. Springer, 2001. ISBN: 9780792374015. URL: https://books.google.it/books?id=XdE4m_xMfL8C.

[11] Srihari Athiyarath, Mousumi Paul, and Srivatsa Krishnaswamy. "A Comparative Study and Analysis of Time Series Forecasting Techniques". In: *SN Computer Science* 1 (2020). ISSN: 2661-8907. DOI: 10.1007/s42979-020-00180-5. URL: https://doi.org/10.1007/s42979-020-00180-5.

[12] Florian Barbieri, Sumedha Rajakaruna, and Arindam Ghosh. "Very short-term photovoltaic power forecasting with cloud modeling: A review". In: *Renewable and Sustainable Energy Reviews* 75 (2017), pp. 242–263. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2016.10.068. URL: https://www.sciencedirect.com/science/article/pii/S136403211630733X.

[13] Jatin Bedi and Durga Toshniwal. "Deep learning framework to forecast electricity demand". In: *Applied Energy* 238 (2019), pp. 1312–1326. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2019.01.113. URL: https://www.sciencedirect.com/science/article/pii/S0306261919301217.

[14] Souhaib Ben Taieb, Gianluca Bontempi, Amir F. Atiya, and Antti Sorjamaa. "A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition". In: *Expert Systems with Applications* 39.8 (2012), pp. 7067–7083. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2012.01.039. URL: https://www.sciencedirect.com/science/article/pii/S0957417412000528.

[15] Christoph Bergmeir and José M. Benítez. "On the use of cross-validation for time series predictor evaluation". In: *Information Sciences* 191 (2012). Data Mining for Software Trustworthiness, pp. 192–213. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2011.12.028. URL: https://www.sciencedirect.com/science/article/pii/S0020025511006773.

[16] Anastasia Borovykh, Sander Bohte, and Cornelis W. Oosterlee. *Conditional Time Series Forecasting with Convolutional Neural Networks*. 2017. DOI: 10.48550/ARXIV.1703.04691. URL: https://arxiv.org/abs/1703.04691.

[17] Lim Bryan and Zohren Stefan. "Time-series forecasting with deep learning: a survey". In: *Philosophical Transactions of the Royal Society A* 379 (2021). ISSN: 1471-2962. DOI: https://doi.org/10.1098/rsta.2020.0209. URL: https://royalsocietypublishing.org/doi/full/10.1098/rsta.2020.0209.

[18] Mengmeng Cai, Manisa Pipattanasomporn, and Saifur Rahman. "Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques". In: *Applied Energy* 236 (2019), pp. 1078–1088. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2018.12.042. URL: https://www.sciencedirect.com/science/article/pii/S0306261918318609.

[19] Lijuan Cao. "Support vector machines experts for time series forecasting". In: *Neurocomputing* 51 (2003), pp. 321–339. ISSN: 0925-2312. DOI: https://doi.org/10.1016/S0925-2312(02)00577-5. URL: https://www.sciencedirect.com/science/article/pii/S0925231202005775.

[20] Vitor Cerqueira, Luis Torgo, and Igor Mozetič. "Evaluating time series forecasting models: an empirical study on performance estimation methods". In: *Machine Learning* 109 (2020), pp. 1997–2028. ISSN: 1573-0565. DOI: 10.1007/s10994-020-05910-7. URL: https://doi.org/10.1007/s10994-020-05910-7.

[21] Wen Chen, Kaile Zhou, Shanlin Yang, and Cheng Wu. "Data quality of electricity consumption data in a smart grid environment". In: *Renewable and Sustainable Energy Reviews* 75 (2017), pp. 98–105. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2016.10.054. URL: https://www.sciencedirect.com/science/article/pii/S1364032116307109.

[22] Yi-Wei Chen, Qingquan Song, and Xia Hu. "Techniques for Automated Machine Learning". In: *SIGKDD Explor. Newsl.* 22.2 (Jan. 2021), pp. 35–50. ISSN: 1931-0145. DOI: 10.1145/3447556.3447567. URL: https://doi.org/10.1145/3447556.3447567.

[23] Ying Chen, Wei Xian Xue, and Xing Long Xie. "Big-Data-Based Modeling of Electricity Consumption Behavior". In: *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*. 2018, pp. 1380–1387. DOI: 10.1109/IAEAC.2018.8577770.

[24] Gopal Chitalia, Manisa Pipattanasomporn, Vishal Garg, and Saifur Rahman. "Robust short-term electrical load forecasting framework for commercial buildings using deep recurrent neural networks". In: *Applied Energy* 278 (2020), p. 115410. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2020.115410. URL: https://www.sciencedirect.com/science/article/pii/S0306261920309223.

[25] Utpal Kumar Das, Kok Soon Tey, Mehdi Seyedmahmoudian, Saad Mekhilef, Moh Yamani Idna Idris, Willem Van Deventer, Bend Horan, and Alex Stojcevski. "Forecasting of photovoltaic power generation and model optimization: A review". In: *Renewable and Sustainable Energy Reviews* 81 (2018), pp. 912–928. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2017.08.017. URL: https://www.sciencedirect.com/science/article/pii/S1364032117311620.

[26] Gabriel de Freitas Viscondi and Solange N. Alves-Souza. "A Systematic Literature Review on big data for solar photovoltaic electricity generation forecasting". In: *Sustainable Energy Technologies and Assessments* 31 (2019), pp. 54–63. ISSN: 2213-1388. DOI: https://doi.org/10.1016/j.seta.2018.11.008. URL: https://www.sciencedirect.com/science/article/pii/S2213138818301036.

[27] Jan G. De Gooijer and Rob J. Hyndman. "25 years of time series forecasting". In: *International Journal of Forecasting* 22.3 (2006). Twenty five years of forecasting, pp. 443–473. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2006.01.001. URL: https://www.sciencedirect.com/science/article/pii/S0169207006000021.

[28] Domingos S. de O. Santos Júnior, João F.L. de Oliveira, and Paulo S.G. de Mattos Neto. "An intelligent hybridization of ARIMA with machine learning models for time series forecasting". In: *Knowledge-Based Systems* 175 (2019), pp. 72–86. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2019.03.011. URL: https://www.sciencedirect.com/science/article/pii/S0950705119301327.

[29] Chirag Deb, Fan Zhang, Junjing Yang, Siew Eang Lee, and Kwok Wei Shah. "A review on time series forecasting techniques for building energy consumption". In: *Renewable and Sustainable Energy Reviews* 74 (2017), pp. 902–924. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2017.02.085. URL: https://www.sciencedirect.com/science/article/pii/S1364032117303155.

[30] Difan Deng, Florian Karl, Frank Hutter, Bernd Bischl, and Marius Lindauer. *Efficient Automated Deep Learning for Time Series Forecasting*. 2022. DOI: 10.48550/ARXIV.2205.05511. URL: https://arxiv.org/abs/2205.05511.

[31] Jianguang Deng and Panida Jirutitijaroen. "Short-term load forecasting using time series analysis: A case study for Singapore". In: *2010 IEEE Conference on Cybernetics and Intelligent Systems*. 2010, pp. 231–236. DOI: 10.1109/ICCIS.2010.5518553.

[32] Ashwini Doke and Madhava Gaikwad. "Survey on Automated Machine Learning (AutoML) and Meta learning". In: *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. 2021, pp. 1–5. DOI: 10.1109/ICCCNT51525.2021.9579526.

[33] Bing Dong, Zhaoxuan Li, S.M. Mahbobur Rahman, and Rolando Vega. "A hybrid model approach for forecasting future residential electricity consumption". In: *Energy and Buildings* 117 (2016), pp. 341–351. ISSN: 0378-7788. DOI: https://doi.org/10.1016/j.enbuild.2015.09.033. URL: https://www.sciencedirect.com/science/article/pii/S0378778815302735.

[34] Pei Du, Jianzhou Wang, Wendong Yang, and Tong Niu. "Multi-step ahead forecasting in electrical power system using a hybrid forecasting system". In: *Renewable Energy* 122 (2018), pp. 533–550. ISSN: 0960-1481. DOI: https://doi.org/10.1016/j.renene.2018.01.113. URL: https://www.sciencedirect.com/science/article/pii/S096014811830123X.

[35] Shengdong Du, Tianrui Li, Yan Yang, and Shi-Jinn Horng. "Multivariate time series forecasting via attention-based encoder-decoder framework". In: *Neurocomputing* 388 (2020), pp. 269–279. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.12.118. URL: https://www.sciencedirect.com/science/article/pii/S0925231220300606.

[36] Salijona Dyrmishi, Radwa Elshawi, and Sherif Sakr. "A Decision Support Framework for AutoML Systems: A Meta-Learning Approach". In: *2019 International Conference on Data Mining Workshops (ICDMW)*. 2019, pp. 97–106. DOI: 10.1109/ICDMW.2019.00025.

[37] Radwa Elshawi, Mohamed Maher, and Sherif Sakr. *Automated Machine Learning: State-of-The-Art and Open Challenges*. 2019. DOI: 10.48550/ARXIV.1906.02287. URL: https://arxiv.org/abs/1906.02287.

[38] Cheng Fan, Jiayuan Wang, Wenjie Gang, and Shenghan Li. "Assessment of deep recurrent neural network-based strategies for short-term building energy predictions". In: *Applied Energy* 236 (2019), pp. 700–710. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2018.12.004. URL: https://www.sciencedirect.com/science/article/pii/S0306261918318221.

[39] Ao Feng, Xuelei Zhang, and Xinyu Song. "Unrestricted Attention May Not Be All You Need-Masked Attention Mechanism Focuses Better on Relevant Parts in Aspect-Based Sentiment Analysis". In: *IEEE Access* 10 (2022), pp. 8518–8528. DOI: 10.1109/ACCESS.2022.3142178.

[40] Luís Ferreira, André Pilastri, Carlos Manuel Martins, Pedro Miguel Pires, and Paulo Cortez. "A Comparison of AutoML Tools for Machine Learning, Deep Learning and XGBoost". In: *2021 International Joint Conference on Neural Networks (IJCNN)*. 2021, pp. 1–8. DOI: 10.1109/IJCNN52387.2021.9534091.

[41] Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. "Auto-Sklearn 2.0: Hands-free AutoML via Meta-Learning". In: (2020). DOI: 10.48550/ARXIV.2007.04074. URL: https://arxiv.org/abs/2007.04074.

[42] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. "Efficient and Robust Automated Machine Learning". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper/2015/file/11d0e6287202fced83f79975ec59a3a6-Paper.pdf.

[43] Mingming Gao, Jianjing Li, Feng Hong, and Dongteng Long. "Day-ahead power forecasting in a large-scale photovoltaic plant based on weather classification using LSTM". In: *Energy* 187 (2019), p. 115838. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2019.07.168. URL: https://www.sciencedirect.com/science/article/pii/S0360544219315105.

[44] Arpad Gellert, Adrian Florea, Ugo Fiore, Francesco Palmieri, and Paolo Zanetti. "A study on forecasting electricity production and consumption in smart cities and factories". In: *International Journal of Information Management* 49 (2019), pp. 546–556. ISSN: 0268-4012. DOI: https://doi.org/10.1016/j.ijinfomgt.2019.01.006. URL: https://www.sciencedirect.com/science/article/pii/S0268401218311368.

[45] Pieter Gijsbers, Erin LeDell, Janek Thomas, Sébastien Poirier, Bernd Bischl, and Joaquin Vanschoren. *An Open Source AutoML Benchmark*. 2019. DOI: 10.48550/ARXIV.1907.00909. URL: https://arxiv.org/abs/1907.00909.

[46] Jake Grigsby, Zhe Wang, and Yanjun Qi. *Long-Range Transformers for Dynamic Spatiotemporal Forecasting*. 2021. DOI: 10.48550/ARXIV.2109.12218. URL: https://arxiv.org/abs/2109.12218.

[47] Weihua He, Yongyun Wu, and Xiaohua Li. "Attention Mechanism for Neural Machine Translation: A survey". In: *2021 IEEE 5th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)*. Vol. 5. 2021, pp. 1485–1489. DOI: 10.1109/ITNEC52019.2021.9586824.

[48] Xin He, Kaiyong Zhao, and Xiaowen Chu. "AutoML: A survey of the state-of-the-art". In: *Knowledge-Based Systems* 212 (2021), p. 106622. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2020.106622. URL: https://www.sciencedirect.com/science/article/pii/S0950705120307516.

[49] Amirreza Heidari and Dolaana Khovalyg. "Short-term energy use prediction of solar-assisted water heating system: Application case of combined attention-based LSTM and time-series decomposition". In: *Solar Energy* 207 (2020), pp. 626–639. ISSN: 0038-092X. DOI: https://doi.org/10.1016/j.solener.2020.07.008. URL: https://www.sciencedirect.com/science/article/pii/S0038092X20307398.

[50] Hansika Hewamalage, Christoph Bergmeir, and Kasun Bandara. "Recurrent Neural Networks for Time Series Forecasting: Current status and future directions". In: *International Journal of Forecasting* 37.1 (2021), pp. 388–427. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2020.06.008. URL: https://www.sciencedirect.com/science/article/pii/S0169207020300996.

[51] Rob J. Hyndman. "A brief history of forecasting competitions". In: *International Journal of Forecasting* 36.1 (2020). M4 Competition, pp. 7–14. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2019.03.015. URL: https://www.sciencedirect.com/science/article/pii/S016920701930086X.

[52] Rob J. Hyndman and Anne B. Koehler. "Another look at measures of forecast accuracy". In: *International Journal of Forecasting* 22.4 (2006), pp. 679–688. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2006.03.001. URL: https://www.sciencedirect.com/science/article/pii/S0169207006000239.

[53] Rich H. Inman, Hugo T.C. Pedro, and Carlos F.M. Coimbra. "Solar forecasting methods for renewable energy integration". In: *Progress in Energy and Combustion Science* 39.6 (2013), pp. 535–576. ISSN: 0360-1285. DOI: https://doi.org/10.1016/j.pecs.2013.06.002. URL: https://www.sciencedirect.com/science/article/pii/S0360128513000294.

[54] Indrajeet Y. Javeri, Mohammadhossein Toutiaee, Ismailcem B. Arpinar, John A. Miller, and Tom W. Miller. "Improving Neural Networks for Time-Series Forecasting using Data Augmentation and AutoML". In: *2021 IEEE Seventh International Conference on Big Data Computing Service and Applications (BigDataService)*. 2021, pp. 1–8. DOI: 10.1109/BigDataService52369.2021.00006.

[55] Mei Jie, Gao Ciwei, Chen Xiao, and Yi Yongxian. "A customer baseline load prediction and optimization method based on non-demand-response factors". In: *2016 China International Conference on Electricity Distribution (CICED)*. 2016, pp. 1–5. DOI: 10.1109/CICED.2016.7576207.

[56] Haifeng Jin, Qingquan Song, and Xia Hu. "Auto-Keras: An Efficient Neural Architecture Search System". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, 1946–1956. ISBN: 9781450362016. DOI: 10.1145/3292500.3330648. URL: https://doi.org/10.1145/3292500.3330648.

[57] Shubhra Kanti Karmaker ("Santu"), Md. Mahadi Hassan, Micah J. Smith, Lei Xu, Chengxiang Zhai, and Kalyan Veeramachaneni. "AutoML to Date and Beyond: Challenges and Opportunities". In: *ACM Comput. Surv.* 54.8 (Oct. 2021). ISSN: 0360-0300. DOI: 10.1145/3470918. URL: https://doi.org/10.1145/3470918.

[58] Kübra Kaysal, Fatih Onur Hocaoğlu, and Nihat Öztürk. "Comparison the Performance of Different Optimization Methods in Artificial Intelligence Based Electricity Production Forecasting". In: *2022 10th International Conference on Smart Grid (icSmartGrid)*. 2022, pp. 236–239. DOI: 10.1109/icSmartGrid55722.2022.9848724.

[59] Junhong Kim, Jihoon Moon, Eenjun Hwang, and Pilsung Kang. "Recurrent inception convolution neural network for multi short-term load forecasting". In: *Energy and Buildings* 194 (2019), pp. 328–341. ISSN: 0378-7788. DOI: https://doi.org/10.1016/j.enbuild.2019.04.034. URL: https://www.sciencedirect.com/science/article/pii/S0378778819308072.

[60] Tae-Young Kim and Sung-Bae Cho. "Predicting residential energy consumption using CNN-LSTM neural networks". In: *Energy* 182 (2019), pp. 72–81. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2019.05.230. URL: https://www.sciencedirect.com/science/article/pii/S0360544219311223.

[61] Taehoon Kim, Dongeun Lee, Jaesik Choi, Anna Spurlock, Alex Sim, Annika Todd, and Kesheng Wu. "Extracting Baseline Electricity Usage Using Gradient Tree Boosting". In: *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*. 2015, pp. 734–741. DOI: 10.1109/SmartCity.2015.156.

[62] Dan Li, Ya Tan, Yuanhang Zhang, Shuwei Miao, and Shuai He. "Probabilistic forecasting method for mid-term hourly load time series based on an improved temporal fusion transformer model". In: *International Journal of Electrical Power & Energy Systems* 146 (2023), p. 108743. ISSN: 0142-0615. DOI: https://doi.org/10.1016/j.ijepes.2022.108743. URL: https://www.sciencedirect.com/science/article/pii/S0142061522007396.

[63] Pengtao Li, Kaile Zhou, Xinhui Lu, and Shanlin Yang. "A hybrid deep learning model for short-term PV power forecasting". In: *Applied Energy* 259 (2020), p. 114216. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2019.114216. URL: https://www.sciencedirect.com/science/article/pii/S0306261919319038.

[64] Youru Li, Zhenfeng Zhu, Deqiang Kong, Hua Han, and Yao Zhao. "EA-LSTM: Evolutionary attention-based LSTM for time series prediction". In: *Knowledge-Based Systems* 181 (2019), p. 104785. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2019.05.028. URL: https://www.sciencedirect.com/science/article/pii/S0950705119302400.

[65] Huanyue Liao and Krishnanand Kaippilly Radhakrishnan. "Short-Term Load Forecasting with Temporal Fusion Transformers for Power Distribution Networks". In: *2022 IEEE Sustainable Power and Energy Conference (iSPEC)*. 2022, pp. 1–5. DOI: 10.1109/iSPEC54162.2022.10033079.

[66] Bryan Lim, Sercan Ö. Arık, Nicolas Loeff, and Tomas Pfister. "Temporal Fusion Transformers for interpretable multi-horizon time series forecasting". In: *International Journal of Forecasting* 37.4 (2021), pp. 1748–1764. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2021.03.012. URL: https://www.sciencedirect.com/science/article/pii/S0169207021000637.

[67] Tao Liu, Zehan Tan, Chengliang Xu, Huanxin Chen, and Zhengfei Li. "Study on deep reinforcement learning techniques for building energy consumption forecasting". In: *Energy and Buildings* 208 (2020), p. 109675. ISSN: 0378-7788. DOI: https://doi.org/10.1016/j.enbuild.2019.109675. URL: https://www.sciencedirect.com/science/article/pii/S0378778819324740.

[68] Yeqi Liu, Chuanyang Gong, Ling Yang, and Yingyi Chen. "DSTP-RNN: A dual-stage two-phase attention-based recurrent neural network for long-term and multivariate time series prediction". In: *Expert Systems with Applications* 143 (2020), p. 113082. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2019.113082. URL: https://www.sciencedirect.com/science/article/pii/S0957417419307997.

[69] Peter Lusis, Kaveh Rajab Khalilpour, Lachlan Andrew, and Ariel Liebman. "Short-term residential load forecasting: Impact of calendar effects and forecast granularity". In: *Applied Energy* 205 (2017), pp. 654–669. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2017.07.114. URL: https://www.sciencedirect.com/science/article/pii/S0306261917309881.

[70] Gonçalo Luís, João Esteves, and Nuno Pinho da Silva. "Energy Forecasting Using an Ensamble of Machine Learning Methods Trained Only with Electricity Data". In: *2020 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*. 2020, pp. 449–453. DOI: 10.1109/ISGT-Europe47291.2020.9248865.

[71] Miguel López Santos, Xela García-Santiago, Fernando Echevarría Camarero, Gonzalo Blázquez Gil, and Pablo Carrasco Ortega. "Application of Temporal Fusion Transformer for Day-Ahead PV Power Forecasting". In: *Energies* 15.14 (2022). ISSN: 1996-1073. DOI: 10.3390/en15145232. URL: https://www.mdpi.com/1996-1073/15/14/5232.

[72] Zhitong Ma, Cantao Ye, and Weibin Ma. "Support vector regression for predicting building energy consumption in southern China". In: *Energy Procedia* 158 (2019). Innovative Solutions for Energy Transitions, pp. 3433–3438. ISSN: 1876-6102. DOI: https://doi.org/10.1016/j.egypro.2019.01.931. URL: https://www.sciencedirect.com/science/article/pii/S1876610219309762.

[73] Joseph Manu. *PMODERN TIME SERIES FORECASTING WITH PYTHON explore industry-ready time series forecasting using modern machine learning and deep learning*. PACKT PUBLISHING LIMITED, 2022. ISBN: 9781803246802.

[74] Ricardo P. Masini, Marcelo C. Medeiros, and Eduardo F. Mendes. "Machine learning advances for time series forecasting". In: *Journal of Economic Surveys* 37.1 (2023), pp. 76–111. DOI: https://doi.org/10.1111/joes.12429. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/joes.12429. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/joes.12429.

[75] A. Mellit, A. Massi Pavan, and V. Lughi. "Deep learning neural networks for short-term photovoltaic power forecasting". In: *Renewable Energy* 172 (2021), pp. 276–288. ISSN: 0960-1481. DOI: https://doi.org/10.1016/j.renene.2021.02.166. URL: https://www.sciencedirect.com/science/article/pii/S0960148121003475.

[76] S. Mirasgedis, Y. Sarafidis, E. Georgopoulou, D.P. Lalas, M. Moschovits, F. Karagiannis, and D. Papakonstantinou. "Models for mid-term electricity demand forecasting incorporating weather influences". In: *Energy* 31.2 (2006), pp. 208–227. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2005.02.016. URL: https://www.sciencedirect.com/science/article/pii/S0360544205000393.

[77] Elena Mocanu, Phuong H. Nguyen, Madeleine Gibescu, and Wil L. Kling. "Deep learning for estimating building energy consumption". In: *Sustainable Energy, Grids and Networks* 6 (2016), pp. 91–99. ISSN: 2352-4677. DOI: https://doi.org/10.1016/j.segan.2016.02.005. URL: https://www.sciencedirect.com/science/article/pii/S2352467716000163.

[78] Shahzad Muzaffar and Afshin Afshari. "Short-Term Load Forecasts Using LSTM Networks". In: *Energy Procedia* 158 (2019). Innovative Solutions for Energy Transitions, pp. 2922–2927. ISSN: 1876-6102. DOI: https://doi.org/10.1016/j.egypro.2019.01.952. URL: https://www.sciencedirect.com/science/article/pii/S1876610219310008.

[79] Thiloshon Nagarajah and Guhanathan Poravi. "A Review on Automated Machine Learning (AutoML) Systems". In: *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*. 2019, pp. 1–6. DOI: 10.1109/I2CT45611.2019.9033810.

[80] Amril Nazir, Abdul Khalique Shaikh, Abdul Salam Shah, and Ashraf Khalil. "Forecasting energy consumption demand of customers in smart grid using Temporal Fusion Transformer (TFT)". In: *Results in Engineering* 17 (2023), p. 100888. ISSN: 2590-1230. DOI: https://doi.org/10.1016/j.rineng.2023.100888. URL: https://www.sciencedirect.com/science/article/pii/S2590123023000154.

[81] Cuong Nguyen and Jeremy J. Roberts. "Green Button Data-Access Model for Smart Cities: Lessons Learned on Security, Transfer, Authorization, and Standards-Compliance in Sharing Energy & Water Usage Data". In: *Proceedings of the 2nd ACM/EIGSCC Symposium on Smart Cities and Communities*. SCC '19. Association for Computing Machinery, 2019. ISBN: 9781450369787. DOI: 10.1145/3357492.3358624. URL: https://doi.org/10.1145/3357492.3358624.

[82] Duc Anh Nguyen, Anna V. Kononova, Stefan Menzel, Bernhard Sendhoff, and Thomas Back. "Efficient AutoML via Combinational Sampling". In: *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. 2021, pp. 01–10. DOI: 10.1109/SSCI50451.2021.9660073.

[83] Zhaoyang Niu, Guoqiang Zhong, and Hui Yu. "A review on the attention mechanism of deep learning". In: *Neurocomputing* 452 (2021), pp. 48–62. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2021.03.091. URL: https://www.sciencedirect.com/science/article/pii/S092523122100477X.

[84] Mariana Oliveira and Luis Torgo. "Ensembles for Time Series Forecasting". In: *Proceedings of the Sixth Asian Conference on Machine Learning*. Ed. by Dinh Phung and Hang Li. Vol. 39. Proceedings of Machine Learning Research. Nha Trang City, Vietnam: PMLR, Nov. 2015, pp. 360–370. URL: https://proceedings.mlr.press/v39/oliveira14.html.

[85] Rafael Pedro and Arlindo L. Oliveira. "Assessing the Impact of Attention and Self-Attention Mechanisms on the Classification of Skin Lesions". In: *2022 International Joint Conference on Neural Networks (IJCNN)*. 2022, pp. 1–8. DOI: 10.1109/IJCNN55064.2022.9892274.

[86] Ratchakit Phetrittikun, Kerdkiat Suvirat, Thanakron Na Pattalung, Chanon Kongkamol, Thammasin Ingviya, and Sitthichok Chaichulee. "Temporal Fusion Transformer for forecasting vital sign trajectories in intensive care patients". In: *2021 13th Biomedical Engineering International Conference (BMEiCON)*. 2021, pp. 1–5. DOI: 10.1109/BMEiCON53485.2021.9745215.

[87] Radu Platon, Vahid Raissi Dehkordi, and Jacques Martel. "Hourly prediction of a building's electricity consumption using case-based reasoning, artificial neural networks and principal component analysis". In: *Energy and Buildings* 92 (2015), pp. 10–18. ISSN: 0378-7788. DOI: https://doi.org/10.1016/j.enbuild.2015.01.047. URL: https://www.sciencedirect.com/science/article/pii/S0378778815000651.

[88] Xi Qi, Lihua Tian, Chen Li, Hui Song, and Jiahui Yan. "Singing Melody Extraction Based on Combined Frequency-Temporal Attention and Attentional Feature Fusion with Self-Attention". In: *2022 IEEE International Symposium on Multimedia (ISM)*. 2022, pp. 220–227. DOI: 10.1109/ISM55400.2022.00050.

[89] V. Sackdara, S. Premrudeepreechacharn, and K. Ngamsanroaj. "Electricity demand forecasting of Electricite Du Lao (EDL) using neural networks". In: *TENCON 2010 - 2010 IEEE Region 10 Conference*. 2010, pp. 640–644. DOI: 10.1109/TENCON.2010.5686767.

[90] Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. "Financial time series forecasting with deep learning : A systematic literature review: 2005-2019". In: *Applied Soft Computing* 90 (2020), p. 106181. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2020.106181. URL: https://www.sciencedirect.com/science/article/pii/S1568494620301216.

[91] Minglei Shao, Xin Wang, Zhen Bu, Xiaobo Chen, and Yuqing Wang. "Prediction of energy consumption in hotel buildings via support vector machines". In: *Sustainable Cities and Society* 57 (2020), p. 102128. ISSN: 2210-6707. DOI: https://doi.org/10.1016/j.scs.2020.102128. URL: https://www.sciencedirect.com/science/article/pii/S2210670720301153.

[92] Zhipeng Shen, Yuanming Zhang, Jiawei Lu, Jun Xu, and Gang Xiao. "A novel time series forecasting model with deep learning". In: *Neurocomputing* 396 (2020), pp. 302–313. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2018.12.084. URL: https://www.sciencedirect.com/science/article/pii/S0925231219304461.

[93] Shun-Yao Shih, Fan-Keng Sun, and Hung-yi Lee. "Temporal pattern attention for multivariate time series forecasting". In: *Machine Learning* 108 (2019), pp. 1421–1441. ISSN: 1573-0565. DOI: 10.1007/s10994-019-05815-0. URL: https://doi.org/10.1007/s10994-019-05815-0.

[94] Arunesh Kumar Singh, S Khatoon Ibraheem, Md Muazzam, and DK Chaturvedi. "An overview of electricity demand forecasting techniques". In: *Network and complex systems* 3.3 (2013), pp. 38–48.

[95] Slawek Smyl. "A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting". In: *International Journal of Forecasting* 36.1 (2020). M4 Competition, pp. 75–85. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2019.03.017. URL: https://www.sciencedirect.com/science/article/pii/S0169207019301153.

[96] Sobrina Sobri, Sam Koohi-Kamali, and Nasrudin Abd. Rahim. "Solar photovoltaic generation forecasting methods: A review". In: *Energy Conversion and Management* 156 (2018), pp. 459–497. ISSN: 0196-8904. DOI: https://doi.org/10.1016/j.enconman.2017.11.019. URL: https://www.sciencedirect.com/science/article/pii/S0196890417310622.

[97] Nivethitha Somu, Gauthama Raman M R, and Krithi Ramamritham. "A hybrid model for building energy consumption forecasting using long short term memory networks". In: *Applied Energy* 261 (2020), p. 114131. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2019.114131. URL: https://www.sciencedirect.com/science/article/pii/S0306261919318185.

[98] Nivethitha Somu, Gauthama Raman M R, and Krithi Ramamritham. "A deep learning framework for building energy consumption forecast". In: *Renewable and Sustainable Energy Reviews* 137 (2021), p. 110591. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2020.110591. URL: https://www.sciencedirect.com/science/article/pii/S1364032120308753.

[99] Evangelos Spiliotis, Andreas Kouloumos, Vassilios Assimakopoulos, and Spyros Makridakis. "Are forecasting competitions data representative of the reality?" In: *International Journal of Forecasting* 36.1 (2020). M4 Competition, pp. 37–53. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2018.12.007. URL: https://www.sciencedirect.com/science/article/pii/S0169207019300159.

[100] Stefano Frizzo Stefenon, Laio Oriel Seman, Viviana Cocco Mariani, and Leandro dos Santos Coelho. "Aggregating Prophet and Seasonal Trend Decomposition for Time Series Forecasting of Italian Electricity Spot Prices". In: *Energies* 16.3 (2023). ISSN: 1996-1073. DOI: 10.3390/en16031371. URL: https://www.mdpi.com/1996-1073/16/3/1371.

[101] James W. Taylor and Roberto Buizza. "Using weather ensemble predictions in electricity demand forecasting". In: *International Journal of Forecasting* 19.1 (2003), pp. 57–70. ISSN: 0169-2070. DOI: https://doi.org/10.1016/S0169-2070(01)00123-6. URL: https://www.sciencedirect.com/science/article/pii/S0169207001001236.

[102] Sean J Taylor and Benjamin Letham. "Forecasting at scale". In: *PeerJ Preprints* 5 (2017), e3190v2. ISSN: 2167-9843. DOI: 10.7287/peerj.preprints.3190v2. URL: https://doi.org/10.7287/peerj.preprints.3190v2.

[103] Ahmed Tealab. "Time series forecasting using artificial neural networks methodologies: A systematic review". In: *Future Computing and Informatics Journal* 3.2 (2018), pp. 334–340. ISSN: 2314-7288. DOI: https://doi.org/10.1016/j.fcij.2018.10.003. URL: https://www.sciencedirect.com/science/article/pii/S2314728817300715.

[104] Alper Tokgöz and Gözde Ünal. "A RNN based time series approach for forecasting turkish electricity load". In: *2018 26th Signal Processing and Communications Applications Conference (SIU)*. 2018, pp. 1–4. DOI: 10.1109/SIU.2018.8404313.

[105] Anh Truong, Austin Walters, Jeremy Goodsitt, Keegan Hines, C. Bayan Bruss, and Reza Farivar. "Towards Automated Machine Learning: Evaluation and Comparison of AutoML Approaches and Tools". In: *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. 2019, pp. 1471–1479. DOI: 10.1109/ICTAI.2019.00209.

[106] Lorenzo Vaccaro, Giuseppe Sansonetti, and Alessandro Micarelli. "An Empirical Review of Automated Machine Learning". In: *Computers* 10.1 (2021). ISSN: 2073-431X. DOI: 10.3390/computers10010011. URL: https://www.mdpi.com/2073-431X/10/1/11.

[107] William VanDeventer, Elmira Jamei, Gokul Sidarth Thirunavukkarasu, Mehdi Seyedmahmoudian, Tey Kok Soon, Ben Horan, Saad Mekhilef, and Alex Stojcevski. "Short-term PV power forecasting using hybrid GASVM technique". In: *Renewable Energy* 140 (2019), pp. 367–379. ISSN: 0960-1481. DOI: https://doi.org/10.1016/j.renene.2019.02.087. URL: https://www.sciencedirect.com/science/article/pii/S0960148119302411.

[108] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2017. DOI: 10.48550/ARXIV.1706.03762. URL: https://arxiv.org/abs/1706.03762.

[109] Huaizhi Wang, Haiyan Yi, Jianchun Peng, Guibin Wang, Yitao Liu, Hui Jiang, and Wenxin Liu. "Deterministic and probabilistic forecasting of photovoltaic power based on deep convolutional neural network". In: *Energy Conversion and Management* 153 (2017), pp. 409–422. ISSN: 0196-8904. DOI: https://doi.org/10.1016/j.enconman.2017.10.008. URL: https://www.sciencedirect.com/science/article/pii/S019689041730910X.

[110] Jian Qi Wang, Yu Du, and Jing Wang. "LSTM based long-term energy consumption prediction with periodicity". In: *Energy* 197 (2020), p. 117197. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2020.117197. URL: https://www.sciencedirect.com/science/article/pii/S0360544220303042.

[111] Kejun Wang, Xiaoxia Qi, and Hongda Liu. "A comparison of day-ahead photovoltaic power forecasting models based on deep learning neural network". In: *Applied Energy* 251 (2019), p. 113315. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2019.113315. URL: https://www.sciencedirect.com/science/article/pii/S0306261919309894.

[112] Kejun Wang, Xiaoxia Qi, and Hongda Liu. "Photovoltaic power forecasting based LSTM-Convolutional Network". In: *Energy* 189 (2019), p. 116225. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2019.116225. URL: https://www.sciencedirect.com/science/article/pii/S0360544219319206.

[113] Ran Wang, Shilei Lu, and Wei Feng. "A novel improved model for building energy consumption prediction based on model integration". In: *Applied Energy* 262 (2020), p. 114561. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2020.114561. URL: https://www.sciencedirect.com/science/article/pii/S0306261920300738.

[114] Yi Wang, Dahua Gan, Mingyang Sun, Ning Zhang, Zongxiang Lu, and Chongqing Kang. "Probabilistic individual load forecasting using pinball loss guided LSTM". In: *Applied Energy* 235 (2019), pp. 10–20. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2018.10.078. URL: https://www.sciencedirect.com/science/article/pii/S0306261918316465.

[115] Lulu Wen, Kaile Zhou, and Shanlin Yang. "Load demand forecasting of residential buildings using a deep learning model". In: *Electric Power Systems Research* 179 (2020), p. 106073. ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2019.106073. URL: https://www.sciencedirect.com/science/article/pii/S037877961930392X.

[116] Tom Wilcox, Nanlin Jin, Peter Flach, and Joshua Thumim. "A Big Data platform for smart meter data analytics". In: *Computers in Industry* 105 (2019), pp. 250–259. ISSN: 0166-3615. DOI: https://doi.org/10.1016/j.compind.2018.12.010. URL: https://www.sciencedirect.com/science/article/pii/S0166361518303749.

[117] Binrong Wu, Lin Wang, and Yu-Rong Zeng. "Interpretable wind speed prediction with multivariate time series and temporal fusion transformers". In: *Energy* 252 (2022), p. 123990. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2022.123990. URL: https://www.sciencedirect.com/science/article/pii/S0360544222008933.

[118] Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. *Deep Transformer Models for Time Series Forecasting: The Influenza Prevalence Case*. 2020. DOI: 10.48550/ARXIV.2001.08317. URL: https://arxiv.org/abs/2001.08317.

[119] Weizhong Yan. "Toward Automatic Time-Series Forecasting Using Neural Networks". In: *IEEE Transactions on Neural Networks and Learning Systems* 23.7 (2012), pp. 1028–1039. DOI: 10.1109/TNNLS.2012.2198074.

[120] Yandong Yang, Weijun Hong, and Shufang Li. "Deep ensemble learning based probabilistic load forecasting in smart grids". In: *Energy* 189 (2019), p. 116324. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2019.116324. URL: https://www.sciencedirect.com/science/article/pii/S0360544219320195.

[121] M. Zamo, O. Mestre, P. Arbogast, and O. Pannekoucke. "A benchmark of statistical regression methods for short-term forecasting of photovoltaic electricity production, part I: Deterministic forecast of hourly production". In: *Solar Energy* 105 (2014), pp. 792–803. ISSN: 0038-092X. DOI: https://doi.org/10.1016/j.solener.2013.12.006. URL: https://www.sciencedirect.com/science/article/pii/S0038092X13005239.

[122] M. Zamo, O. Mestre, P. Arbogast, and O. Pannekoucke. "A benchmark of statistical regression methods for short-term forecasting of photovoltaic electricity production. Part II: Probabilistic forecast of daily production". In: *Solar Energy* 105 (2014), pp. 804–816. ISSN: 0038-092X. DOI: https://doi.org/10.1016/j.solener.2014.03.026. URL: https://www.sciencedirect.com/science/article/pii/S0038092X14001601.

[123] Haixiang Zang, Ruiqi Xu, Lilin Cheng, Tao Ding, Ling Liu, Zhinong Wei, and Guoqiang Sun. "Residential load forecasting based on LSTM fusing self-attention mechanism with pooling". In: *Energy* 229 (2021), p. 120682. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2021.120682. URL: https://www.sciencedirect.com/science/article/pii/S0360544221009312.

[124] G.Peter Zhang. "Time series forecasting using a hybrid ARIMA and neural network model". In: *Neurocomputing* 50 (2003), pp. 159–175. ISSN: 0925-2312. DOI: https://doi.org/10.1016/S0925-2312(01)00702-0. URL: https://www.sciencedirect.com/science/article/pii/S0925231201007020.

[125] Hao Zhang, Yajie Zou, Xiaoxue Yang, and Hang Yang. "A temporal fusion transformer for short-term freeway traffic speed multistep prediction". In: *Neurocomputing* 500 (2022), pp. 329–340. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2022.05.083. URL: https://www.sciencedirect.com/science/article/pii/S092523122200666X.

[126] Hai Zhong, Jiajun Wang, Hongjie Jia, Yunfei Mu, and Shilei Lv. "Vector field-based support vector regression for building energy consumption prediction". In: *Applied Energy* 242 (2019), pp. 403–414. ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2019.03.078. URL: https://www.sciencedirect.com/science/article/pii/S0306261919304878.

[127] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. *Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting*. 2020. DOI: 10.48550/ARXIV.2012.07436. URL: https://arxiv.org/abs/2012.07436.

[128] Yi Zhou, Nanrun Zhou, Lihua Gong, and Minlin Jiang. "Prediction of photovoltaic power output based on similar day analysis, genetic algorithm and extreme learning machine". In: *Energy* 204 (2020), p. 117894. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2020.117894. URL: https://www.sciencedirect.com/science/article/pii/S036054422031001X.

[129] Lucas Zimmer, Marius Lindauer, and Frank Hutter. *Auto-PyTorch Tabular: Multi-Fidelity MetaLearning for Efficient and Robust AutoDL*. 2020. DOI: 10.48550/ARXIV.2006.13799. URL: https://arxiv.org/abs/2006.13799.

[130] Miguel A. Zúñiga-García, G. Santamaría-Bonfil, G. Arroyo-Figueroa, and Rafael Batres. "An Association-Rule Method for Short-Term Electricity Demand Forecasting and Consumption Pattern Recognition". In: *2018 Seventeenth Mexican International Conference on Artificial Intelligence (MICAI)*. 2018, pp. 3–7. DOI: 10.1109/MICAI46078.2018.00008.

[131] Zeynep Çamurdan and Murat Can Ganiz. "Machine learning based electricity demand forecasting". In: *2017 International Conference on Computer Science and Engineering (UBMK)*. 2017, pp. 412–417. DOI: 10.1109/UBMK.2017.8093428.