Forecasting for Solar Power Farms

By

Tiago Mourão Pires

Master Thesis in Modelling, Data Analysis
and Decision Support Systems

Supervised by:

Professor João Manuel Portela da Gama

Maria Isabel Pinto Preto

Faculdade de Economia

Universidade do Porto

2022

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The Importance of Renewable Energy

The ever growing appetite of the world population for energy, to fuel its social and economic development, has put an increasing stress on Earth climate system due to the emissions of greenhouse gases originated in the larger part from the use of fossil fuel energy sources such as coal, oil and gas. In addition to these environmental concerns, geopolitical constraints have also been a factor to consider in the security and sustainability of the energy supply.

This is the background, described by Owusu and Asumadu-Sarkodie (2016), where alternative sources of energy such as hydro, geothermal, wind, ocean or solar have gained prominence as means to satisfy energy demand. Solar in particular can be used in different ways, either to produce thermal energy or using the photovoltaic (PV) technology to produce electricity. These renewable energy sources (RES) are considered to be clean in the sense that they don't produce greenhouse gas emissions, only generate minimal waste and can be a mitigation for the global emissions keeping the global warming within reasonable limits.

The European Union roadmap and strategy addressing sustainable development, climate targets and environment goals is depicted in the European Green Deal reviewed by Wolf, Teitge, Mielke, Schütze, and Jaeger (2021) and Fetting (2020). The main goal of this policy is to achieve a net carbon neutral European Union by 2050 while decoupling economic growth, essential for a prosperous society, from natural resources use. Within this framework, some key areas are identified, highlighting the need to supply clean, affordable and secure energy as specially relevant for this work. The need for a transition to clean energy sources is recognized with smart technology as a key enabler to smart grids, hydrogen and other renewable energies.

The decarbonization of energy is a key block towards reaching the climate objectives in 2050 considering that the production and utilization of energy accounts for more than 75% of greenhouse gas emissions in EU. The vision is then for an electrical sector based on renewable sources with the fast phase out of coal and gas, used for complement in the transition.

The Value of Forecast

One of the main barriers to the general deployment of RES, according to Anees (2012), is their variable nature, determined by the weather, which makes power grid integration a very challenging task to keep a reliable and cost-effective supply; this is different behavior to traditional fossil fuel based generators that have the ability to adjust their output based on market incentives. Such behavior can originate power fluctuations on the network operation, instability leading to fault conditions or unacceptable voltage fluctuations in weaker nodes of the grid.

Additionally, in a free-market context, the value of variable renewable energy (VRE) sources is determined by three different aspects summarized by Hirth (2013): the supply is variable and its value depends on the time of generation; the primary resource (wind, sun) is bound to some locations so transmission and distribution constrains mean its value depend on the generation location; finally, the output is uncertain until produced although plants commit to that production the day before delivery. The latter remark is the main driver for this work because it means that forecast errors from VRE power plants need to be balanced and compensated for by the grid operator at short notice, typically by using backup generators which are costly thus reducing the market value of renewable energy.

The aforementioned study highlights predictability and improved forecast models for photovoltaic production in solar power plants as an actual benefit which is recognized by the market as smaller penalties, due to the fact that less balancing is needed. In a basic perspective, the producer communicates the 24-hour ahead production to the market operator; in the final market settlement, the producer will be rewarded for the energy sold and penalized for the difference between expected and real energy delivered. Producers that have newly commissioned solar farms on their portfolio expect to have accurate power forecast from day one. Effectively, power forecasts are used to ensure the efficient management of the electric grid as well as in power trading operations, making this problem particularly interesting and pressing, namely on the accuracy of forecasting results. On one hand, forecast is crucial for energy market operations where producers are penalized by deviations related to the difference between what they expect to produce and what they actually delivered. On the other hand, power forecast is also an important tool for anomaly, drift or fault detection in the elements of photovoltaic installations as shown by Leva, Mussetta, and Ogliari (2018). It is also useful to ensure appropriate (predictive) maintenance and to guarantee the optimization of the deployment of operational teams to the field.

Solar photovoltaic forecast models have traditionally been based on mathematical modeling of the physical components but recent improvements in computational power has enabled data-driven approaches leveraged by machine learning algorithms. While these approaches have found success, building a PV forecast model for a power plant that is just starting its operation, i.e., without representative historical production data records, remains an illusive problem often referred to as a cold-start problem.

Goals of the Thesis

In the context of this thesis, the (sparse) known information about each plant is composed of the farm location, the installed power of PV generators and the electrical characteristics of the panels and inverters. Additionally, data originated in similar PV power plants is also available for the purpose of the study, opening an interesting possibility to evaluate the applicability of the transfer learning paradigm in order to improve the quality of the forecast thus resulting in economical benefit to the producer. For this work, the goal is to develop 10-days ahead hourly production forecasts.

A final note to mention that the work described in this thesis is developed in collaboration with Smartwatt, a Portuguese company that develops artificial intelligence, engineering and monitoring systems aiming for energy resource optimization. The construction and validation of the forecast model was supported with data provided by Smartwatt from photovoltaic power plants currently in operation.

Chapter 2

Literary Review

2.1 Forecast Models

The forecast models typically used for prediction tasks are generally classified in two main groups: physical models and data-driven models. Recently, new approaches have been proposed, effectively combining these two types of models into hybrid models as shown by Xiao, Xu, He, and Sha (2022).

2.1.1 Physical Models

The physical models are based on the comprehensive mathematical modeling and analysis of the physical processes responsible for some phenomenon. These models are built in a very detailed manner describing all physical inputs and relations of the system, typically yielding high accuracy predictions. Due to this explicit coding of physical mechanisms through equations, physical models are also known as white-box models which is connected to the transparency of the underlying estimates. The biggest disadvantage of this type of model consists in the fact that they rely on very accurate information about a big amount of system parameters which are often inaccessible; this inherent complexity often makes the development slow and simulations computationally expensive according to Wei et al. (2018).

A physical model for the electrical production of a PV power plant is proposed by Stanev and Tanev (2018) where a mathematical model is built based on a structural approach including the PV generator, DC link and inverter; the results are validated and accuracy is evaluated against measured data from a real PV plant.

Ma, Yang, and Lu (2014) present different mathematical models to characterize the equivalent circuit of photovoltaic devices, from the cell to module and array. The ideal model doesn't take into account the effect of internal series and parallel resistances thus not being suitable for real-world modeling being seen instead as a step towards more complex models. One and two-diode models are also presented. The one-diode model, with 5 parameters modeling the internal resistances, was selected as a compromise between accuracy and required computational power. The parameters of the model were estimated in MATLAB using the Levenberg-Marquardt algorithm. The results were validated against field data considering three cases of sunny, semi-cloudy and cloudy day using the RMSE and Mean Bias Error (MBE) metrics.

Another mathematical model for the power generation through photovoltaic panels is proposed by Hassan, Jaszczur, and Przenzak (2017). This work describes the calculation of solar radiation components (based on the time of the day, longitude and accounting for the obliquity and eccentricity of Earth's orbit), extraterrestrial solar radiation on an horizontal surface (based on latitude, declination and solar constant radiation), diffuse radiation (considering the division of total radiation in direct and diffuse components through the clearness index) and the incident solar radiation on PV array (accounting for the slope of the PV model, ground reflectance or albedo, the anisotropy index and ratio of beam radiation on tilted to horizontal surface considering the surface azimuth angle). The total solar radiation incident on the surface of the PV module, calculated through the steps described before, is then used to predict the electrical power generated considering the rated capacity of the module and a derate factor which takes into account additional loss factors like snow, electrical losses or shading for instance.

Another example, motivated by the high cost of collecting wind data, is presented by Al-Yahyai, Charabi, and Gastli (2010) showing the development of Numerical Weather Prediction (NWP) models for wind resource assessment; these models typically solve the equations characterizing the physical processes of the atmosphere determining their evolution through time, if the initial condition is known.

This type of models is also used in the context of building energy consumption prediction, for instance by Wei et al. (2018): the model includes all the details regarding the building construction, the thermal characteristics, ventilation, air conditioning systems, occupancy and all other that are deemed relevant to predict the overall consumption of electricity.

2.1.2 Data-Driven Models

With the growing deployment of real-time measuring systems and sensors, data-driven models have been proposed to tackle forecast problems by using historical data (of the target variable and sometimes also other relevant features) to generalize a relationship between the inputs and the outputs with no regard for the underlying physical process. This simplicity constitutes both an advantage when it comes to the speed of development but it also produces models that are less explainable thus called black-box models.

According to Bourdeau, Zhai, Nefzaoui, Guo, and Chatellier (2019), these models are developed in three generic stages where three different datasets, containing historical data, are used. The first one consists in training the algorithm to adjust its parameters to optimize the fit with the train set. Next, the validation dataset is used to provide an unbiased evaluation of the algorithm while tuning the model hyperparameters and possibly enforcing some variable selection. Using a different dataset will help to reduce the risk of overfitting the model, a phenomenon that occurs when the model performs well on the train set but then fails to produce good predictions with other sets. Finally, the test stage provides an unbiased evaluation of the modeling and forecast power of the algorithm and no tuning of model occurs. The description of this process highlights one of the downfalls of this type of models which occurs when there's not enough historical data which will be detrimental in the accuracy of the algorithm; this situation is common in the initial stage of operation of a building or another equipment.

Data-driven models can be developed with two main lines. The first is a statistical approach based on regression techniques, linear or multivariate, like autoregressive integrated moving average (ARIMA). The second approach is based on machine learning (ML) methods which can include algorithms such as artificial neural networks (ANN), support vector machines (SVM), decision trees (DT), random forest, classification and regression trees (CART) and deep learning like long-short term memory networks (LSTM). Wang and Srinivasan (2017) propose a distinction between single (use single algorithm for forecast process) and ensemble methods (combination of different algorithms to handle their weaknesses and strengths).

Monteiro, Fernandez-Jimenez, Ramirez-Rosado, Muñoz-Jimenez, and Lara-Santillan (2013) provide a comparison of two different short-term statistical forecasting models applied to the hourly production of photovoltaic power plants with both using past production data and NWP forecasts at the plant location as input. Then, the first model is based on multiplicative decomposition, adjusting the clear sky irradiation (deterministic component) with a PV production attenuation index calculated from the weather variables forecast. The second model uses a multilayer perceptron neural network (MLP) as base model which is optimized using a genetic algorithm (GA) to determine the input variables (from all the possibilities provided by the NWP model) and also to determine the number of neurons in the hidden layers and the backpropagation training algorithm parameters. Both models, although distinct in nature, achieve similar results for day-ahead predictions thus stressing their applicability to define offer bids for the PV production in the spot electricity market while contributing to the profitability of the PV park by reducing the economic penalties originated by the deviations between bids and real production.

Sala et al. (2019) benchmark different algorithms in the problem of PV forecast: Linear regression, Lasso regression (forces feature selection penalizing features with smaller impact on forecast error), Random Forest, MLP, KNN regressor and LSTM. The results, evaluated with MAE, RMSE, MBE and R^2, show that Lasso and Linear regression outperform more flexible algorithms in short-term horizons; on the other hand, LSTM algorithm seems more suited when considering longer time into the past for the prediction. Some caution is advised on generalizing the best algorithm since PV forecast depends on many variables such as horizon, granularity, input data or geographical location for instance.

2.1.3 Hybrid Models

Hybrid models are often referred to as grey-box models since they combine physical white-box with data-driven black-box models in order to produce forecasts; in some particular scenarios, where the physical simulation is too complex or the input data is not enough for the learning process, it's mentioned by Xiao et al. (2022) that these models can provide better results.

The merge between physical and data-driven models can follow three distinct strategies listed by Foucquier, Robert, Suard, Stéphan, and Jay (2013). The first one has the largest focus on the physical model using the data to estimate variables of the system that are either unknown or not accessible. Secondly, the data approach can be used to simplify the typical complexity of physical systems. Finally, the most flexible composition uses the physical and data-driven component in different places of the model, depending on the suitability of each one; Xiao et al. (2022) cites three ways to achieve this: sequence (using the models in order with the solution of the first one used as input to the second), parallel (the order to use models is not relevant, both will compound the final solution) and feedback (one model has the main role to produce solutions while the second model helps to modify it).

The biggest advantages of hybrid models is that they allow considering limited amount of input data, may use only a rough physical description of the system and retain the physical explainability for the results; on the downside, computational time may be large to compute both approaches and some difficulties emerge when mixing two distinct scientific domains.

This kind of models is applicable for instance in the problem of predicting short-term wind speed where, according to Giebel et al. (2011), the physical models (trying to hold physical considerations as long as possible to reach the best possible estimate) can be used in conjunction to a statistical approach in order to produce successful forecasts. It's concluded that the best approaches are grey-box models where some knowledge of the wind power properties can be used to tune the models to the specific domain or site.

The potential of these models is further demonstrated by Ferreira, Santos, and Lucio (2019) where two different statistical models are used to produce the mean hourly wind speed in two different locations, one coastal and another with more complex topographic features. The first one is based in Holt-Winters model which contains four smoothing elements (level, linear, trend and seasonal) and a random error component; these are estimated through the exponential (weighted arithmetic mean with weights decreasing into the past) smoothing (less variations in time series) method. This model can come both in multiplicative and additive form; the two were used to avoid negative values without physical meaning in the forecast speeds. The second is based on an ANN trained with the backpropagation algorithm to adjust the weights of the network that optimize the error between forecast and observed values. Finally, an hybrid approach was also implemented, using R package *forecastHybrid*, that resulted in the combination of three models: a univariate neural network, a model based on Seasonal and Trend decomposition using Loess (STL) and a exponential smoothing model with Box-Cox transformation, Autoregressive Moving Average (ARMA) and seasonal/rend components. The hybrid approach proved to have the best fit with the time series capturing the variability in the diurnal cycle satisfactorily.

Methods	**Training Data**	**Physical Interpretation**
Physical	No data required for development; may be used for validation.	Results can be interpreted in physical terms.
Statistical	A large amount of training data, collected over an exhaustive and representative period of time, is typically required.	There are several difficulties to interpret results in physical terms.
Hybrid	A small amount of training data, collected over a short period of time, is usually sufficient.	Results can be interpreted in physical terms.

Table 2.1: Summary of the characteristics of each model type adapted from Giebel et al. (2011).

2.2 Cold Start Problem

The rise of Artificial Intelligence (AI) techniques for the development of forecast models has put in evidence the so-called cold start problem. In fact, these methods generally use big datasets in order to build the forecast models meaning that there's a difficult challenge in creating a model when suitable data is not available. Formally, the cold start problem occurs in computer-based information systems that require data as forecast model input, when such system cannot build the required inferences due to the scarce information about users or items, according to Jihoon, Junhong, Pilsung, and Eenjun (2020). In other words, the generic cold start problem is related with the sparsity of information that is available to the learning algorithm.

In the context of photovoltaic power production, this problem is relevant to forecast electricity production in new solar farms where enough historical and relevant data hasn't been acquired thus representing an obstacle to the deployment of regular data-driven forecast models.

2.2.1 Examples

Lika, Kolomvatsos, and Hadjiefthymiades (2014) show that the area of recommender systems is a very important application facing the cold start problem due to the difficulty in making recommendations for new users or items without the relevant historical background to guide the solution. In this context, the problem can include recommendations for new users, recommendations for new items or recommendations of new items to new users. This paper proposes the integration of a classification task in the commonly used collaborative filter approach, leveraging demographic features to identify users with similar behaviors.

Aguilar, Munoz-Romero, and Rojo-Álvarez (2020) deal with a cold start forecast problem in the supply chain area, related to promotional sales where there's few evidence to support the modeling with traditional ML models; additionally, interpretability is also a key aspect since it can affect the usability of the prediction. The idea is that the proposed method can include feature selection (to find the drivers of sales) and then select the closest products (neighbours) in order to produce a contrastive explanation of the results.

Xie, Tank, Greaves-Tunnell, and Fox (2017) address the issue of cold start problem in a time series context where long range forecasts are required. The proposed model combines a regression component (leveraging the external features of data, also known as metadata) and a matrix factorization term (exploring the structure contained in the patterns shared across periods and series). Additionally, the problem of warm start is also considered, dealing with how the cold start predictions can be affected as new observations are collected.

This problem is also relevant in the area of energy demand prediction. Florian (2020), for instance, introduces the problem of load nowcasting: in the context of electrical grid operation, the system operators publish real-time data just after power delivery in the form of preliminary values, deduced from limited metered data, that deviate from the final metered values. Nowcasting aims at providing more accurate preliminary values for the load, using only the limited data from the very recent past.

2.3 Transfer Learning

2.3.1 Definition

The success and accuracy of traditional machine learning algorithms is strongly tied to the availability of large datasets representing some phenomenon that is to be modeled; the idea is that a larger dataset will provide a better inductive process in extracting representative patterns that can be useful for the prediction task. In the real world however, it's often costly, time-consuming or simply not feasible to gather such amounts of labeled data. To address this scarceness of data, new fields of research were explored and summarized by Zhuang et al. (2020).

Transfer Learning (TL) is a psychology-inspired machine learning paradigm based on the generalization theory of transfer which explains how knowledge can be transferred across domains: knowledge gained on a related or source domain can be deployed to improved the learning performance in a target domain. A typical example is how someone who has learned riding a bike will have a faster learning on how to ride a motorcycle.

Using the formal definitions, as explained by Fan et al. (2020), a domain D is composed of a feature space X and their marginal probabilities $P(X)$ being thus denoted as:

$$D = \{X, P(X)\}$$

A task T also includes the label Y and $P(Y|X)$ the conditional probability of Y given X:

$$T = \{Y, P(Y|X)\}$$

The transfer learning process ends in the domain D_t and target task T_t by determining the target conditional probability distribution $P(Y_t|X_t)$, using the knowledge transferred from the source domain D_s and task T_s.

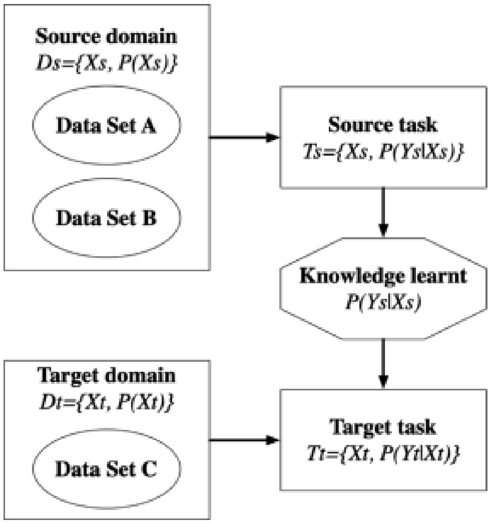


Figure 2.1: Basic scheme of transfer learning according to Fan et al. (2020).

An interesting aspect of TL, reviewed by Pan and Yang (2009) and Weiss, Khoshgoftaar, and Wang (2016), is that the transference of knowledge doesn't necessarily bring a positive impact on the target domain thus being termed negative transfer. This can be caused by the matching relevance between source and target domains harming the capacity of the algorithm to find the generalizable knowledge across domains. It has been shown by Rosenstein, Marx, Kaelbling, and Dietterich (2005) that if the source and target tasks are too different, the enforcement of transfer learning will prejudice the performance of the target task. This highlights the necessity of measuring the task-relatedness or having some transferability measure to guarantee the possibility of generalization of each source domain to other learning scenarios.

2.3.2 Classification

Since transfer learning is a quite diverse range of methods and paradigms, there are several distinct classification of TL approaches based on different criteria.

According to the Settings of Source and Target Domains and Tasks

Pan and Yang (2009) synthesizes three main groups according to the settings of source and target domain tasks: inductive transfer learning (source and target domains are the same, tasks are different), transductive transfer learning (source and target domains are not the same but they have the same tasks) and unsupervised transfer learning (source and target share the domains but tasks are different yet related).

According to the Task Feature Space

Weiss et al. (2016) focus on the distinction between homogeneous and heterogeneous transfer learning, taking into account the difference between domains: while homogeneous TL include domains sharing the same feature space (that may differ on marginal distributions), heterogeneous TL occurs when the domains don't have the same feature space. In the latter type of TL, it's common to mention domain adaption as the range of techniques employed to increase the similarity between source and target.

According to the Implementation Methods

Another common division, also mentioned by Pan and Yang (2009), is more concerned with how transfer learning is implemented, referring to instance-based TL (re-weighting some parts of source data to use in the learning task of the target domain), feature representation-based TL (finding a good feature representation that minimizes the difference between source and target domains), parameter-based TL (source and target tasks sharing some model hyperparameters) and relational knowledge-based TL (some of the relations within data in source and task domain is similar).

This work will focus and detail on this classification and the methods used in literature to implement transfer learning approaches on machine learning tasks.

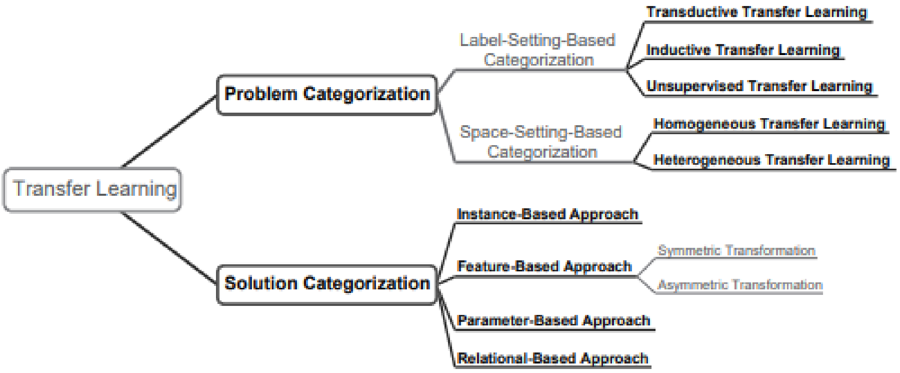

Transfer Learning
Problem Categorization
Label-Setting-Based Categorization
Transductive Transfer Learning
Inductive Transfer Learning
Unsupervised Transfer Learning
Space-Setting-Based Categorization
Homogeneous Transfer Learning
Heterogeneous Transfer Learning
Solution Categorization
Instance-Based Approach
Feature-Based Approach
Symmetric Transformation
Asymmetric Transformation
Parameter-Based Approach
Relational-Based Approach

Figure 2.2: Classification of transfer learning paradigms according to Zhuang et al. (2020).

2.3.3 Implementation Methods

Instance-based

Data-based interpretation of transfer learning emphasizes the possibility of transferring knowledge between source and target domain by the transformation and conditioning of data thus minimizing the disparity in the distribution between domains and instances.

The instance weighting strategy allow to transfer knowledge between a source domain with a big number of labeled observations into a target domain with limited observations, assuming that both with differ only in marginal distributions. This procedure would enable the exploitation of labeled instance of the source domain in order to improve the performance in task domain. Jiang and Zhai (2007) propose some adaption heuristics based on instance weighting such as removal of misleading observations in the source domain, add more weight to target than source labeled instances or increase train set with target instances containing predicted labels.

Feature-based

The feature transformation strategy rely on the changing some original feature into a new feature representation thus minimizing marginal and conditional distributions difference (measured through maximum mean discrepancy (MMD) for instance) while keeping the structures and properties of data. There are three main subdivisions of this strategy regarding features: augmentation, reduction and alignment.

Feature augmentation can consist of simple replication of features (note the feature augmentation method (FAM) proposed by Daumé III (2009)) or other approaches based on feature stacking. Some methods of feature extraction are also mentioned by Zhuang et al. (2020): feature mapping aiming at minimizing MMD (instead of the focus on variance of techniques such as Principal Component Analysis (PCA) in regular ML) and feature clustering that looks for a more abstract representation for the original features. Feature selection is also proposed to select the pivot features of a dataset, defined as the ones that behave similarly in both domains being thus appropriate as a vehicle for knowledge transfer.

Feature encoding is also specially relevant in the deep learning and stacking architectures; the encoder creates a more abstract form of the input while the decode maps back that representation minimizing the reconstruction error.

Finally, feature alignment focuses on the implicit statistic or spectral features of the dataset instead of the explicit ones.

Parameter-based

Looking away from data, another major possibility for transfer learning is to use model-level regularizers to the learner objective function so that the knowledge of pre-trained models based on source domain will be transferred to target domain on the train step. Duan, Xu, and Tsang (2012) proposes a generic paradigm called Domain Adaptation Machine where pre-trained classifiers are obtained from labeled data of either domains; then, based on the smoothness assumption, a domain dependent regularizer forces that the base and new target classifiers will have similar predicted values on the unlabeled observations of target domain.

Another strategy uses the parameters of the learned models. The simplest idea consists in parameter sharing which is popular in ANN based solutions since it's possible to learn a complete network from source task data and then retrain the last selected layers using the target task data in a fine-tuning process. Parameter restriction is slightly different from sharing because it simply enforces that parameters will be similar between source and target tasks, not that they actually share the same values. The construction of an ensemble model, already existing in regular ML, can also be applied to TL: a group of candidate classifiers is learned in source domain tasks and then applied iteratively over the labeled target instances in order to select the classifiers that will be ensemble to generate the final predictions.

Finally, it has been found, for instance by Ghifary, Kleijn, and Zhang (2014), that deep learning based algorithms outperform shallow ANN or Support Vector Machine (SVM) models when it comes to dealing with the domain adaption problem. Consequentially, deep learning techniques are popular in the transfer learning paradigm, either in a reconstruction-based (autoencoders) or discrepancy-based approach (using measures such as MMD). There are also some approaches, such as this application by J. Li et al. (2020), that use adversarial transfer learning for fault detection in a fault category previously not existing in either source or target domain.

Relational-based

A lesser explored type of TL, covered by Kumaraswamy, Odom, Kersting, Leake, and Natarajan (2015), uses relational models or some other representations like graphs in order to achieve domain independent transfer. The basic idea is to represent the relational structure between multiple objects to capture source domain knowledge and transfer it to the target domain, either through implicit or explicit mapping of the relational structure across domains. The results of a demonstration case study, contained in the same paper, support the idea that relational-based TL can achieve performances similar or better than other state-of-the-art TL methods.

2.3.4 Applications of Transfer Learning to Forecast

The transfer learning paradigm is used in a broad range of problems that goes from medicine, transports, recommender-systems and many others presented by Zhuang et al. (2020). This report however will focus on applications that, due to their nature, can provide a better insight into the photovoltaic forecast problem, either by sharing some of the features, patterns or ML models better suited to the energy production prediction area.

Power Production

Zhou, Zhou, Mao, and Xi (2020) propose a transfer learning paradigm for photovoltaic power production as way to circumvent the problem faced by data-driven methods on PV plants without a large historical background of observations. The idea is to transfer knowledge from the solar radiation (affected by solar energy in upper atmosphere, the atmospheric attenuation and local cloud disturbance) since it's a closely related task sharing similar distributions. A LSTM model is proposed to capture the periodicity with its memory capabilities in order to make day-ahead predictions with a frequency of 10 minutes meaning 144 values. The Bayesian optimization method was used for hyperparameter optimization. Transfer learning is implemented in typical fashion by pre-training the model in the source domain and then the weights of the final layer are tuned using target data. The evaluation metrics are absolute percentage error (APE), MAPE and RMSE with the model being benchmarked against its standard version with the TL step. The results show transfer learning to be specially valuable with insufficient data meaning that its advantage declines as more data is gradually obtained.

Hu, Zhang, and Zhou (2016) apply a transfer learning paradigm to the problem of short-term wind speed prediction, relevant for wind farm control, but often difficult to solve with methods like NWP (unavailable or spatially insufficient) or data-driven models (without enough historical background). TL transfers the knowledge from other wind farms using deep neural networks to extract patterns which are then tuned with the from target farms. This paper uses a shared-hidden-layer DNN architecture where the hidden layers are shared between source and target domain with the output layers being different. The results are assessed from different points of view using MAE, MSE, RMSE and MAPE; in general, the proposed strategy improves the performance on scarce data scenarios but tends to lose importance as the size of target train set increases.

Electrical Load

Seung-Min et al. (2020) use transfer learning for monthly electric load forecast, crucial in power grid operation. This is a problem with typically insufficient data points since, by definition, only one is generated per month. This experiment uses Pearson correlation coefficient to select relevant domains for the target task thus improving the efficiency of transfer learning. The TL models using DNN fine-tuned with target domain data are benchmarked against Multiple Linear Regression (ML), Random Forest, XGB (Extreme Gradient Boosting) and DNN using the MAPE and normalized root mean squared error (NRMSE) metrics; the results show that the TL models outperform both the regular ML methods and the basic form of DNN without TL.

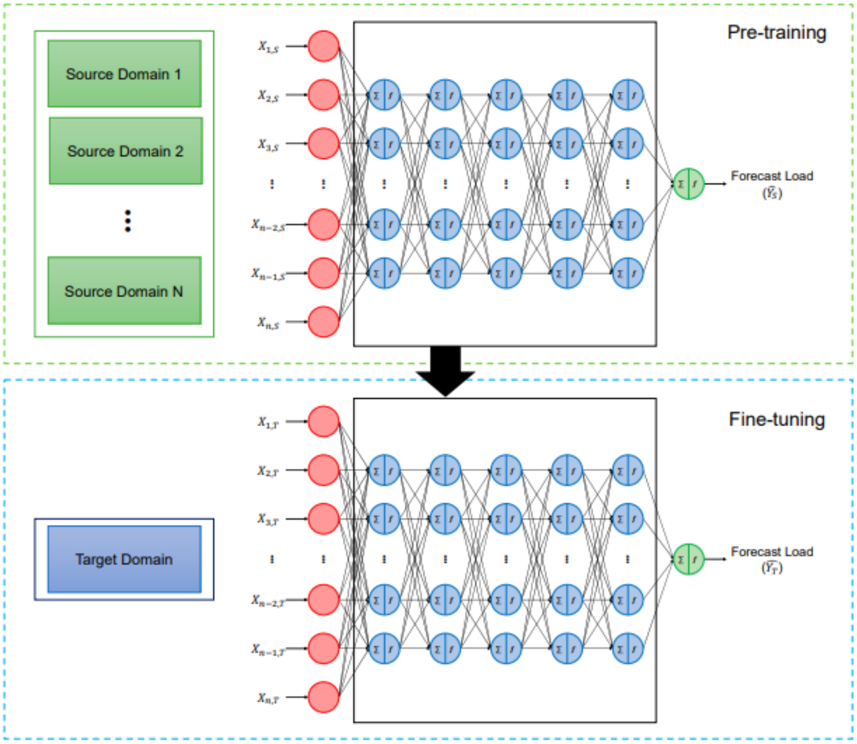


Figure 2.3: Schematics of Transfer Learning implementation based on Deep Neural Network from Seung-Min et al. (2020).

Zhang and Luo (2015) also focus the problem of load prediction for cities, relevant in a smart grid context. The paper proposes a source task selection algorithm to avoid negative knowledge transfer and then a Gaussian Process (GP) model with TL for the predictions, benchmarked against standard GP, Autoregressive and Support Vector Regression based models. The results, evaluated with normalized mean square error (NMSE) highlight the better performance of TL algorithm and the negative transfer avoided by considering nearby cities (source task selection).

Hooshmand and Sharma (2019) present a case study for load consumption where the day-ahead hourly load is predicted using 4 weeks of historical data on the assumption that it is representative of trends and seasonality patterns related with daily and weekly periods. The ML model consists of a convolutional network where the knowledge is transferred using the scarce target data to adjust the weights of the final layers thus being a case of model-based TL. The results, evaluated with MAE, show that TL coupled with CNN will outperform the standard CNN but also a benchmark SARIMA model.

Building Energy Consumption

Fan et al. (2020) address the problem of short-term building energy consumption based on data-driven methods which is often not possible either due to having a new building or simply the fact that most buildings don't have sufficient data coming from the monitoring systems. Transfer learning is then used to leverage the knowledge obtained from buildings with effective measurement systems in order to forecast the energy consumption in the next 24 hours. The paper explore a network-based approach to transfer learning based on deep learning with three main blocks: 1D convolutional layers (to obtain temporal features from the time series), then recurrent layers based on LSTM units are used to acquire the interaction between temporal features and thirdly the categorical variables are turned numeric using one-hot encoding. Regularization techniques are used to avoid overfitting and the parameters of the model are optimized with grid-search. Two different strategies for TL are tested: use pre-trained model to extract features (take whole pre-trained fixed parameters except for the output layer) or use the pre-trained model for weight initialization which are tuned with target data. The results are evaluated with common metric such as RMSE (root mean squared error), CV-RMSE (coefficient of variation of the root mean squared error) and PIR (performance improvement ratio). The results show that transfer learning has a positive effect on RMSE which reduces by 67% in the weight initialization transfer learning scenario.

A. Li, Xiao, Fan, and Hu (2021) also take into account a similar problem presenting an ANN-based solution with transfer learning for one-hour ahead prediction based on previous 24 hours consumption. The paper explain the use of a backpropagation neural network (BPNN) with three layers: the first hidden layer with 24 nodes, the second with 12 codes. Rectified Linear Unit was selected as activation function. Transfer learning is applied by using the source domain for weight initialization, fine-tuned afterwards with target data. The evaluation metrics selected were Mean Absolute Percentage Error (MAPE) and Mean Square Error (MSE). The conclusions are not just that TL increases prediction accuracy but also that improvement is bigger if the available dataset is smaller.

Jihoon et al. (2020) describe an approach using tree-based machine learning (ML) methods such as Multivariate Random Forest (MRF) or Random Forest (RF) to solve the cold-start problem in the context of short-term load forecast of building energy consumption. The models are combined to consider the different electricity consumption patterns typical from working days and holidays. The results show an improvement in MAPE, RMSE and MAE error metrics.

Abdulrahman et al. (2021) frame the residential buildings energy consumption problem and lists the most typically used base models including ANN, Deep Belief Network (DBN), Recurrent Neural Network (RNN), LSTM, Elman Neural Network (ENN), Nonlinear Autoregressive Neural Network (NARX), MLP and Convolutional Neural Network (CNN). Then, the application of transfer learning coupled with LSTM is discussed for medium to long term consumption forecast.

Gao, Ruan, Fang, and Yin (2020) use a combination of deep and transfer learning methods to boost the accuracy in the forecast of energy consumption for building with small amounts of historical information. The paper compares the performance of three models: LSTM to allow the information to be memorized for longer time and transmitted along the time sequence (also used without transfer learning for baseline performance), seq2seq model with two layers of LSTM as encoder and decoder and a 2D CNN which is effective for feature extraction, building high-level features automatically. Sequential TL was employed with all data from source domain used for pre-training and one month coming from target used for fine-tuning. Mean Absolute Error (MAE), MAPE and CV-RMSE were used to evaluate the results; both models were able to improve those metrics by 20-30%.

Fang et al. (2021) depict a hybrid deep learning model for short-term prediction with limited historical data. It uses a LSTM feature extractor (to acquire temporal features across source and target domain) and a domain adversarial neural network (DANN) to find domain invariant features through adversarial domain adaptation. The idea is that the model trained with source data can be applied directly to target without worsening performance due to domain mismatch. The LSTM is a RNN that adds internal memory and a gate mechanism to the base RNN allow to capture longer term dependence better; it also includes a forget fate to manage how historical data should be discarded or not. The results are evaluated against LSTM trained with data from target, source or both without domain adaption using MSE, MAE, MAPE and CV-RMSE; the hybrid LSTM-DANN model improve these metrics around 15% while also outperforming variations of the model where LSTM was replaced by a CNN or fully connected layer.

Ribeiro, Grolinger, ElYamany, Higashino, and Capretz (2018) present Hephaestus, a parameter and instance-based transfer learning method for building energy consumption forecast which, unlike traditional TL methods, considers the effect of seasonality in the domains; it's based on time series multi-feature regression considering seasonal and trend adjustments. This is particularly relevant when using source datasets from different buildings with different distributions and seasonal profiles. The method is meant to be deployed in the pre and post-processing stages of the typical ML pipeline meaning that it's algorithm independent. It's divided in four stages: time series adaptation (seasonality and trend effect effects removed and transferred to target), non-temporal domain adaptation (invariant features), appropriate machine learning algorithm and finally adjustment (prediction affected by the factors derived in the two first stages). A case study is proposed to test the method using MAPE and MSE to conclude that improvements can reach up to 11% against a scenario without it.

2.3.5 Related Concepts

Semi-Supervised Learning (SSL), according to Chapelle, Schölkopf, and Zien (2006), lies somewhere in between supervised and unsupervised learning methods using a mixture of completely unlabeled with some labeled observations. Such an approach is known to be effective specially because all data comes from the same distribution. Additionally, some assumptions usually are implicit in SSL algorithms; the main ones are the smoothness (when two observations are close in a high density region of the input space, the outputs should also be close), the cluster (if two examples are grouped in the same cluster, their class should be the same) and the manifold assumptions (high dimensional data lies on a low dimensional manifold). While both SSL and TL share these assumptions, they differ in the fact that while in SSL both labeled and unlabeled observations come from the same distribution, in TL the distributions of source and target domains are typically not the same.

Multi-View Learning (MVL) is based on the idea that different views of the same object can be represented by different feature sets resulting in more information to be used by the ML algorithm thus leading to an enhanced performance. These approaches are typically classified, according to Xu, Tao, and Xu (2013), as co-training, subspace learning or multiple kernel learning. MVL concepts can be used as a way of ensuring knowledge transfer across domains in some applications.

Multi-Task Learning (MTL) is based on the core idea that if there's a joint learning process including related (but not identical) tasks, it's possible to use the knowledge contained in all tasks, thus improving the learning performance; the problem of small data is worked around by explore information from related tasks. As mentioned by Zhang and Yang (2018), this paradigm stresses the importance of two factors: the task relatedness and the definition of task. MTL and TL share similar modeling strategies and techniques but they differ in the sense that TL focus more on target than source task while MTL is focused simultaneously in all related tasks.

Reference	Task	Model	Transfer Learning	Dataset	Horizon
A. Li et al. (2021)	Building energy	Data-Driven (BPNN)	Parameter-based (Final layers tuning)	Consumption Weather	1h (1h)
Fan et al. (2020)	Building energy	Data-Driven (LSTM)	Parameter-based (Feature extraction and final layers tuning)	Building metadata Consumption Weather	24h (1h)
Ribeiro et al. (2018)	Building energy	Data-Driven (Season. Adjust.)	Parameter-based Instance-based	Consumption Weather	1 month (1 day)
Fang et al. (2021)	Building energy	Data-Driven (LSTM-DANN)	Feature-based (Feature extraction and domain adaptation)	Consumption Weather	1 week (1 day)
Gao et al. (2020)	Building energy	Data-Driven (LSTM/CNN)	Parameter-based (Final layers tuning)	Consumption by category	1 month (1 day)
Florian (2020)	Electricity load	Data-Driven (Multi-Linear)	-	Load	1 day (15min)
Seung-Min et al. (2020)	Electricity load	Data-Driven (DNN)	Parameter-based (Final layers tuning)	Load Weather Demographic	2 years (1 month)
Jihoon et al. (2020)	Electricity load	Data-Driven (MRF)	Instance-based (Similarity measures)	Load	1 day (1h)
Hooshmand and Sharma (2019)	Electricity load	Data-Driven (CNN)	Parameter-based (Final layers tuning)	Load	1 day (1h)
Zhou et al. (2020)	PV Production	Data-Driven (LSTM)	Parameter-based (Final layers tuning)	Irradiance PV Production	1 week (10min)
Ma et al. (2014)	PV Production	Physical	-	PV Production	1 day (5min)
Sala et al. (2019)	PV Production	Data-Driven (Various, LSTM)	-	PV Production	1 hour (5min)
Stanev and Tanev (2018)	PV Production	Physical	-	PV Production	- (1h)
Hassan et al. (2017)	PV Production	Physical	-	PV Production	- (-)
Monteiro et al. (2013)	PV Production	Data-Driven (MLP)	-	Irradiance Weather PV Production	4 days (1h)
Xie et al. (2017)	Time Series	Data-Driven (Various)	-	Flu Trends Wikipedia Traffic	1 year (1 day/week)
Hu et al. (2016)	Wind Speed	Data-Driven (DNN)	Parameter-based (Final layers tuning)	Wind	8h (10min)
Ferreira et al. (2019)	Wind Speed	Hybrid (ANN, HW)	-	Wind	1 day (1h)

Table 2.2: Summary of time-series forecast models presented in the Literary Review.

2.4 Tools

2.4.1 Data Extraction

The data concerning photovoltaic production will be provided on CSV or XLS format with no foreseen automated method to access and download it for the purposes of this work. Regarding meteorological data, as mentioned before, it will be downloaded from MeteoGalicia through XML-based communication using R package *meteoForecast* to interact with THREDDS (Thematic Realtime Environmental Distributed Data Service) and obtain operation modeling data for weather research forecast.

An evaluation of the data pipeline was be conducted and, due to the complexity and sizable amount data, it was decided to build an automated process based on R where data could be downloaded from the sources and stored in appropriate database (DB) structures. The outline and details of this system are described in 3.1. The combination of R and a database system provided an automated, fast, robust and error-free process for data extraction and integration.

2.4.2 Modeling

Due to its flexibility, readability, simplicity and open-source nature, Python was selected as the programming language to be used for the data pre and post-processing and also for the modeling and development of a machine learning algorithm to solve the photovoltaic production forecast problem.

Using Python, some different libraries and frameworks will be deployed and tested before making a final decision for the model. A typical library used in Python for classification or regression is *scikit-learn* as described by Pedregosa et al. (2011). Another option consists in using *TensorFlow*, presented by Abadi et al. (2016), enabling the optimization and training of algorithms for a wide range of applications including deep learning which is a commonly used paradigm as shown in previous section. The *Keras* library, proposed by Chollet (2021), is built over primary deep learning platforms like *TensorFlow* making it another solid option. Finally, *H2O AutoML*, described by LeDell and Poirier (2020), was another open source machine learning platform considered in order to take advantage of the possibility of producing a large number of models in short time, select or stack according to some performance metric.

PVLib library was another toolbox, firstly developed in MATLAB but also available in Python, that was considered for open source, reliable and benchmark implementation of the performance modeling of PV systems. It contains clear sky modeling capabilities and the possibility to acquire meteorological forecast data from models such as Global Forecast System (GFS). The physical model used in this software is based in the single-diode equation mentioned previously in this review. According to Gurupira and Rix (2016), PVLib has shown consistent performance against industry benchmark and commercially available software like PVSyst.

Chapter 3

Data Description and Analysis

3.1 Data Workflow

3.1.1 PV Production

The production data from the photovoltaic power plants used in this work was provided by Smartwatt under a confidentiality agreement due to its commercially sensitive nature; six different sites are considered in this work, five in the southern part of the country and another one in the center. The dataset includes the location of solar farms in Portugal, characterized by an unique ID, the nominal installed power of the PV generators in MW, current operation status and geographical location (latitude and longitude). This information is summarized in table 3.1 and it was stored in a specific database (DB) table called *sites*.

ID	Nominal Power [MW]	Start	End	Total	Granularity
1	0,6	01/01/2020 00:00	30/10/2021 22:45	1 years, 9 months, 29 days	15min
2	12,0	01/01/2020 00:00	31/10/2021 23:45	1 years, 9 months, 30 days	15min
3	49,5	01/01/2020 00:00	31/10/2021 23:45	1 years, 9 months, 30 days	15min
4	5,0	01/01/2020 00:00	31/10/2021 23:45	1 years, 9 months, 30 days	15min
5	36,0	05/10/2020 23:00	31/01/2022 23:45	1 years, 3 months, 26 days	15min
1023	43,6	01/01/2018 00:00	18/10/2021 23:00	3 years, 9 months, 17 days	1h

Table 3.1: Information about the data available for each solar power farm.

For each of the plants, a time series of the power production, in MW, is provided. This is either hourly or quarter-hourly data thus corresponding to 24 (1 hour granularity) or 96 observations (15 minutes granularity) per day with notable exceptions on the two days of each year where Daylight Saving Time starts or ends or eventual missing values that can occur due to sensor malfunction or other technical reasons. The time series contains a first column with the timestamp (with the date and hour of the observation in UTC - coordinated universal time) and the second with the production value. All the data is stored in a DB table called *production*.

It's also relevant to highlight the different length of the time series available for each park. This occurs due to the operation start date being different for each location thus resulting in different amounts of collected data. That is actually one of the focus of this work, the prediction of

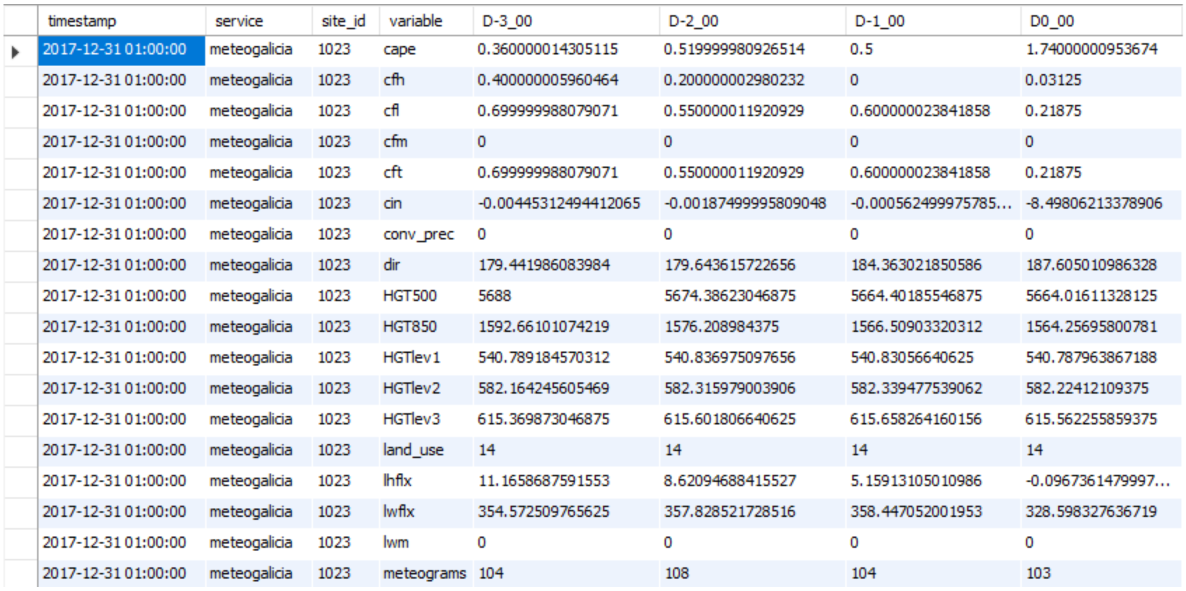
PV production when there's insufficient historical records that regular machine learning models, without transfer learning, typically use for the forecast.

3.1.2 Meteorology

Numerical Weather Prediction models are used as source of meteorological data, retrieved using R software, developed by R Core Team (2020), through the *meteoForecast* package created by Perpiñán and Almeida (2021). The aforementioned package allows to access the outputs of different NWP services for some location; currently, it's possible to get data from models such as GFS (Global Forecast System), MeteoGalicia or NAM (North American Mesoscale Forecast System). For this thesis, MeteoGalicia will be the primary source due to its geographical proximity with the PV parks for which production data is available.

The data obtained from MeteoGalicia consists of hourly data describing atmospheric, physical and geographical features of relevance for the meteorological forecast. The typical format consists in the following columns: a timestamp (with the date and hour of the observation in UTC time), the forecast value for some variable three days before the timestamp (D-3), D-2 forecast, D-1 forecast, D-0 forecast, the variable identifier and the geographical coordinates (latitude and longitude). The spatial resolution is an input defined when using the Meteo Galicia data: for this work a 4 kilometers raster resolution was used when collecting data. This also constitutes a relative advantage when compared to another meteorological models such as GFS or NAM.

There are 45 features available for download including, for instance, cloud cover at different levels, wind speed, gust and direction, visibility, snow level, temperature at different levels, air pressure and many others; all this data is stored in a DB table called *meteo*, for each site.



	timestamp	service	site_id	variable	D-3_00	D-2_00	D-1_00	D0_00
▶	2017-12-31 01:00:00	meteogalicia	1023	cape	0.360000014305115	0.519999980926514	0.5	1.74000000953674
	2017-12-31 01:00:00	meteogalicia	1023	cfh	0.400000005960464	0.200000002980232	0	0.03125
	2017-12-31 01:00:00	meteogalicia	1023	cfl	0.699999988079071	0.550000011920929	0.600000023841858	0.21875
	2017-12-31 01:00:00	meteogalicia	1023	cfm	0	0	0	0
	2017-12-31 01:00:00	meteogalicia	1023	cft	0.699999988079071	0.550000011920929	0.600000023841858	0.21875
	2017-12-31 01:00:00	meteogalicia	1023	cin	-0.00445312494412065	-0.00187499995809048	-0.000562499975785...	-8.49806213378906
	2017-12-31 01:00:00	meteogalicia	1023	conv_prec	0	0	0	0
	2017-12-31 01:00:00	meteogalicia	1023	dir	179.441986083984	179.643615722656	184.363021850586	187.605010986328
	2017-12-31 01:00:00	meteogalicia	1023	HGT500	5688	5674.38623046875	5664.40185546875	5664.01611328125
	2017-12-31 01:00:00	meteogalicia	1023	HGT850	1592.66101074219	1576.208984375	1566.50903320312	1564.25695800781
	2017-12-31 01:00:00	meteogalicia	1023	HGTlev1	540.789184570312	540.836975097656	540.83056640625	540.787963867188
	2017-12-31 01:00:00	meteogalicia	1023	HGTlev2	582.164245605469	582.315979003906	582.339477539062	582.22412109375
	2017-12-31 01:00:00	meteogalicia	1023	HGTlev3	615.369873046875	615.601806640625	615.658264160156	615.562255859375
	2017-12-31 01:00:00	meteogalicia	1023	land_use	14	14	14	14
	2017-12-31 01:00:00	meteogalicia	1023	lhflx	11.1658687591553	8.62094688415527	5.15913105010986	-0.0967361479997...
	2017-12-31 01:00:00	meteogalicia	1023	lwflx	354.572509765625	357.828521728516	358.447052001953	328.598327636719
	2017-12-31 01:00:00	meteogalicia	1023	lwm	0	0	0	0
	2017-12-31 01:00:00	meteogalicia	1023	meteograms	104	108	104	103

Figure 3.1: MySQL Workbench view of *meteo* table built with Meteo Galicia data.

3.1.3 Raw data integration

In order to handle the large amount of data in a uniform, fast and reliable way, a database system was put in place. The implementation of the mySQL database roughly follows the data sources structure doing a bare minimum of operations in data, storing it as raw as possible to keep traceability. As described in the previous sections, tables were created for power production and meteorological time-series data and then also site-specific data for each solar park and metadata to characterize the variables obtained from the meteorological service.

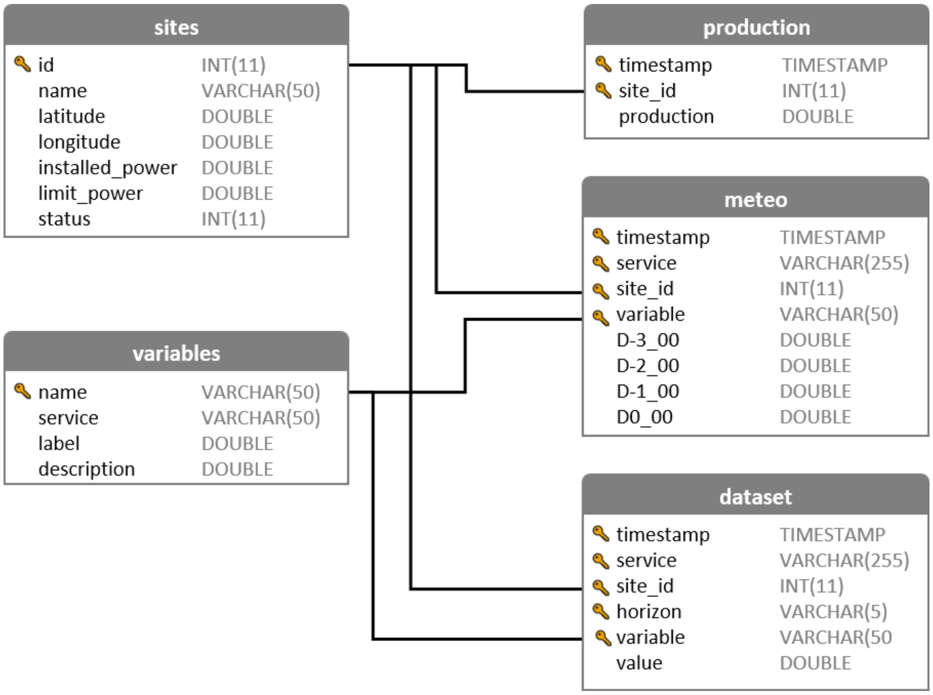


Figure 3.2: Schema of the database used to store all work-related data.

Due to the use of *meteoForecast* package, R programming language was used to manage the entire data pipeline and workflow thus avoiding the coordination with other techniques for extraction and DB storage of data. A data collection script was developed in order to fulfill the following tasks:

- Read solar production data from Excel format files and insert into DB table *production*.
- Retrieve site-specific information from database (latitude, longitude, production time span).
- Download data from MeteoGalicia web service, iterating for all sites and relevant spans.
- Insert meteorological data into DB table *meteo* with the appropriate formats.

After the raw data is stored into mySQL database, R was also used for a merging step, schematized in figure 3.2, where meteorological and production data are brought together in the same *dataset* table. This was facilitated by SQL features such as the possibility to create views on the existing tables; such mechanism was particularly important to aggregate/uniform production data into hourly data matching the time steps of meteo data. Then, it was possible to merge both meteo (for all time horizons) and production tables. The choice of 1h as base for all data was done for three different reasons: readability of results since the forecast can either be interpreted as average power during the hour (units MW) or the amount of energy produced in one hour (units MWh); matching the source used to collect meteorological data; the electricity markets is based on buying and selling energy for each hour.

Finally, R was used to read from this data view created in mySQL and insert it back in a table. This step is particularly relevant for performance issues because SQL views can be quite slow to read from because they require that all associated queries run in the background, any time the view is accessed; on the other hand, it's much faster to read from a table because there are no queries running to create it, only a query to select a subset of static data.

This highlights the role of R as an extraction, transformation and integration tool bringing together the flexibility of web services acting as a bridge to the robustness and storage capabilities of a database system. In short, R and mySQL together, linked as depicted in figure 3.3, provided a strong backbone of data management to build the forecast model from.

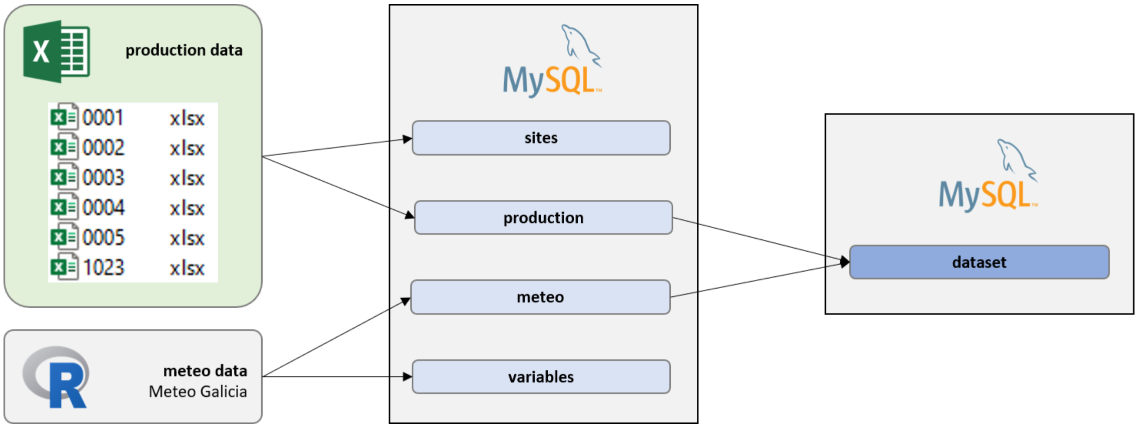
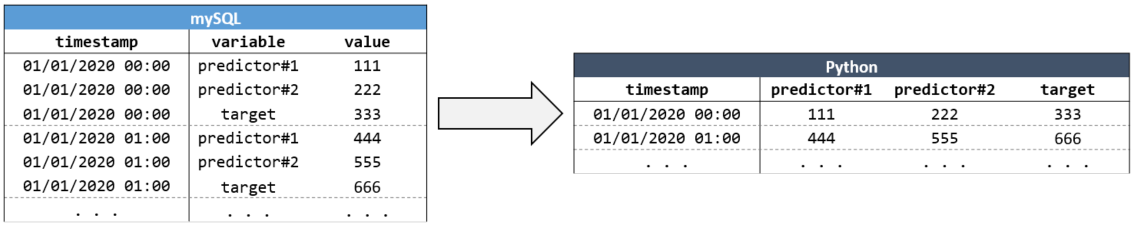


Figure 3.3: Summarized data integration workflow.

3.1.4 From Database to Python dataset structure

While R and SQL were used for the data integration part of this work, Python was the preferred language for the forecast algorithm implementation due to the free libraries available in the area of machine learning. So, the first step to create the forecast model was to read data into Python using the *pandas* and *sqlalchemy* libraries. It should be noted that no filtering was done on the SQL query; instead, all Meteo Galicia forecast time horizons were read into Python and used for different purposes which will be explained in later sections.

The storage of data in mySQL follows the typical best practices keeping the number of columns in *dataset* table to a minimum. This means that for each timestamp, there are many different rows characterized by a different variable (predictors and target), site or horizon (days ahead from Meteo Galicia forecast). This is in stark contrast with the typical dataset format with each row characterizing a single and unique timestamp with each predictor representing a column. To achieve this format, a Python function was created with the output being a dataframe with the timestamp as first column, target as last column and all the predictor variables in between: the transformation is illustrated in figure 3.4.



mySQL		
timestamp	variable	value
01/01/2020 00:00	predictor#1	111
01/01/2020 00:00	predictor#2	222
01/01/2020 00:00	target	333
01/01/2020 01:00	predictor#1	444
01/01/2020 01:00	predictor#2	555
01/01/2020 01:00	target	666
. . .	. . .	. . .

Python			
timestamp	predictor#1	predictor#2	target
01/01/2020 00:00	111	222	333
01/01/2020 01:00	444	555	666
. . .	. . .	. . .	. . .

Figure 3.4: Data transformation from mySQL to standard dataset format in Python.

3.2 Exploratory Data Analysis

3.2.1 Variables Type

The dataset built using the processes described in the previous chapter contains 1 timestamp variable, 43 predictor variables describing the meteorological features of the time series and 1 target variable corresponding to the hourly solar power production in the park. The full description, with the information of each variable and corresponding units, is detailed in appendix A. The variables are mostly numerical, as presented in table 3.2, with some specific characteristics discussed in the following sections.

Type	Count
Timestamp	1
Numeric	41
Categorical	3
Total	45

Table 3.2: Types and count of available meteorological and solar production variables.

Regarding the missing values for each variable, there is a missing not at random phenomena related with the way database handles the insertion of the Summer time start hour. Nevertheless, no other data is missing and, overall, missing data represents less than 1% of the entire dataset. Due to this very low amount of missing data, no further measures were explored to handle it.

3.2.2 Categorical Variables

One relevant aspect is that 2 of the 3 categorical variables are not really time dependent characteristics related with meteorological events; in fact, they represent simple labels returned by the Meteo Galicia data service. Specifically, *land_use* relates with the land use or vegetation type and *lwm* is a simple binary variable to represent whether the specified point is located of land or water. The other categorical attribute, *meteograms*, contains 16 distinct values that are a reference to graphical representations of meteo data.

Taking into account the nature of the categorical attributes described above, they are unsuited to be used for time series forecast of the target variables thus not being addressed any further in this work.

3.2.3 Numerical Variables

Due to the circumstances discussed in previous section, the dataset can be seen as fully numerical. Then, two variables stand out due to being constant with one distinct value for each site: *weasd* is the water equivalent of accumulated snow depth which is zero across all sites; *topo* is the altitude for each site defined by latitude and longitude. Due to the particular nature of these variables, both can be discarded from the subsequent analysis since they don't contain any time dependent feature that could be helpful for power production forecast.

Univariate Analysis

The complete study of the variables characteristics, for each site, is detailed in appendix B where the target variable (power production) is represented as *y*. This analysis is divided into four main areas: counters (with the count of missing and distinct values for each variable), location (mean, minimum, maximum and quartiles), dispersion (standard deviation, variance and coefficient of variation) and shape measures (skewness, kurtosis).

Predictor Variables

Location Measures

From the analysis of location measures, it's possible to check that variables such as *cfh*, *cfl*, *cfm* and *cft*, which represent the cloud cover of the sky at different levels, are expressed in percentage, thus varying from 0 to 1. The same happens with relative humidity, *rh* variable.

The mean value of the cloud cover variables (*cft*, *cfl* or *cfm*) is generally higher in site 1023. This may be interpreted with the empirical knowledge that typically sunnier weather is more likely to be experienced further in the south of the country, a fact evidenced by lower average cloud covers. This may also explain the differences in *visibility* observed across sites.

Some of the most relevant variables for solar power production relate with the flux of energy between the atmosphere and the Earth, expressed in energy over time and area with corresponding units W/m^2. Sensible and latent heat flux (*shflx* and *lhflx*) relate mostly with heat transfer with the atmosphere. Thus, both these variables contain negative values that can be interpreted as a night time phenomenon where the Earth is cooling, thus releasing energy to the atmosphere in infrared form. Then, it's also worth to take a closer look to *lwflx*, the surface downwelling longwave flux, that takes only positive values because it represents the thermal irradiance reaching the surface in the thermal infrared spectrum, thus being also present during the night. Finally, and likely a very good predictor for solar power production, the surface downwelling shortwave flux, *swflx*, contains a direct and a diffuse component of the solar beam and so it makes sense to have minimum value zero (at night) and a maximum value in the order of magnitude of the solar constant - 1361W/m^2 - which can be defined the energy received from the Sun on a perpendicular surface to its rays without considering any atmospheric effects. It's also very important to stress that photovoltaic cells target the conversion of high energy photons (short-wave side of solar spectrum) - this technical detail further underlines the potential of *swflx* for the forecast model. The relation between the surface downwelling shortwave flux and power production is highlighted in figure 3.5 where both variables exhibit very similar patterns.

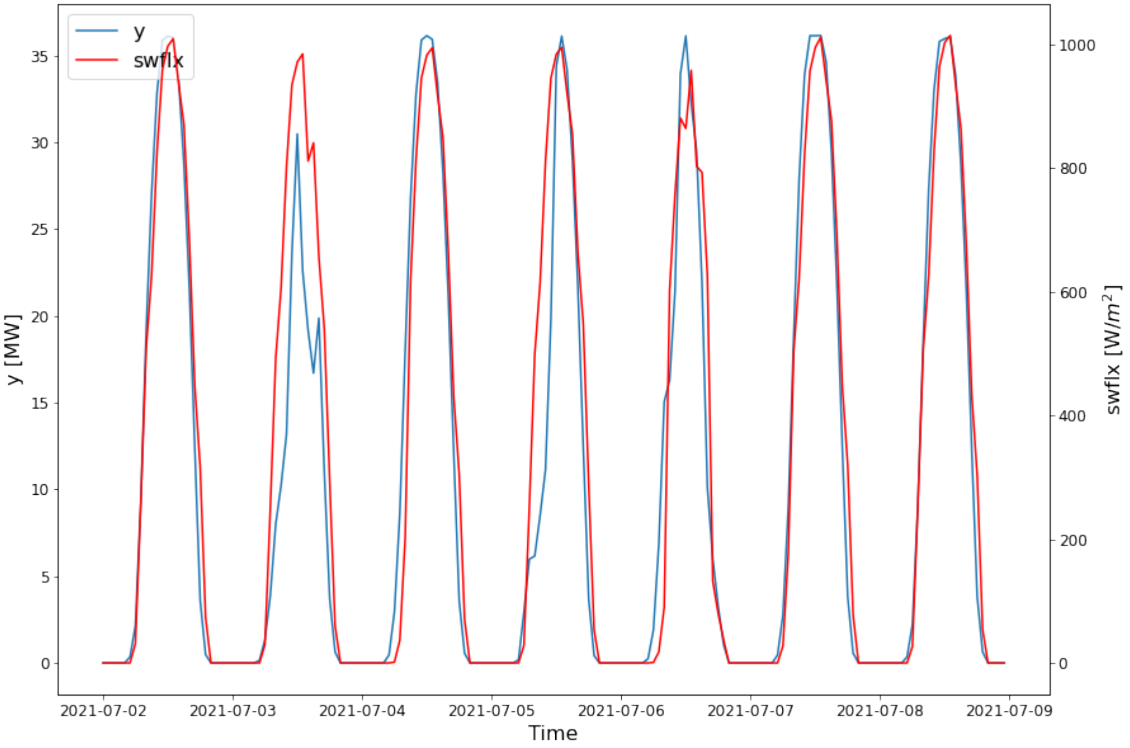


Figure 3.5: Solar power production and *swflx* for a selected week in site 5.

There's also variables related with wind speed such as *v*, *vlev1*, *vlev2*, *vlev3*, *u*, *ulev1*, *ulev2* orr*ulev3* containing negative values; although not explicitly stated in the documentation of the R package used to retrieve data, this is related with the definition of axis and directions for the wind speed vector: *v* refers to the meridional velocity (component of horizontal wind towards north) and *u* refers to the zonal velocity (component of horizontal wind towards east).

Another notable mention is the zero mean for *snow_prec* variables in all sites. In site 1023, the value is slightly above but still approximately zero; this is probably related with the fact that snow is quite rare overall in Portugal but, being further north, this site would be the most likely to have some data different from zero on that specific attribute. The *snowlevel* variable can have a similar explanation exhibiting mean values around 2000m meaning that at the sites location, snow would only be expected at very large heights.

Dispersion Measures

Given the underlying physical units for each variable, the interpretation of any metric should take that into account. Thus it's specially interesting to look into the coefficient of variation for dispersion analysis as this metric is independent from the measure scale. Looking at this measure, the five variables related to the geopotential height (height above sea level of some pressure level) stand out having a variation below 5% for all sites. Also the temperature variables, *temp*, *sst*, *T500* and *T850*, exhibit a similar characteristic; nevertheless, this should be taken carefully because in Kelvin scale, 1 degree difference corresponds to 1 Celsius but the absolute temperature is measured from absolute zero (-273 Celsius) meaning that a small coefficient of variation is to be expected and the variables still can be quite meaningful for the forecast model.

Another highlight of the analysis were the variables related with wind speed, cloud cover and precipitation that typically exhibit the largest values for the coefficient of variation across sites.

Shape Measures

These metrics aim at quantifying the geometric properties of the variable distribution. One standout is the fact that precipitation variables (*prec* and *conv_prec*) have the largest positive skews, meaning that the most frequent values are smaller than the mean; *visibility*, on the contrary, has one of the largest negative skew meaning that most frequent values are larger than the mean.

The same precipitation variables are also highlighted looking at kurtosis excess; in fact, they present the highest positive values, meaning that the tails of the distribution have less weight relative to a normal distribution. On the opposite, variables such as the wind direction, *dir*, have the largest negative kurtosis values with the tails of the distribution being heavier than normal distribution.

Target Variable

The solar power production over time for each site, the target variable, has some particular characteristics that can be discussed separately. First, it's trivial to note that production will never be less than zero (either during the night or day period) and also it cannot exceed the nominal power production capacity of the equipment. It's then expected that minimum value for the variable will be zero and the maximum should roughly correspond to installed capacity. This is exactly what happens across all sites with the notable exception of 1023 where the maximum value registered is 5927 MW which compares to an installed power of 43 MW; such observation should be flagged as an outlier since the underlying physical system doesn't have the capacity to produce such power.

The histograms of the target variable, depicted in figure 3.6 for different sites, don't include the zero production values, in order to get a better visual reading. Showing zero production would highlight the positive skew which should be expected considering that for all the night hours, there is no production, shifting the distribution towards that value. Additionally, both during sunrise and sunset hours, lower production is to be expected since photovoltaic systems are usually mounted in a way to maximize production during peak hours, around noon, thus being less productive on early morning and late afternoon which explains the weight in the left tail of the distribution. On the other hand, the weight of the right tail (corresponding to production close to nominal capacity) is also higher; such pattern does make sense because the design of a solar power farm will aim to optimize the installed power production capacity or, in other words, the economic feasibility of this kind of projects depends on running the plant closer to nominal capacity when possible. Combined, these two effects are responsible for negative kurtosis calculated across sites for the target variable. The only exception is site 1023, due to outliers as described previously, thus being kept out of this analysis.

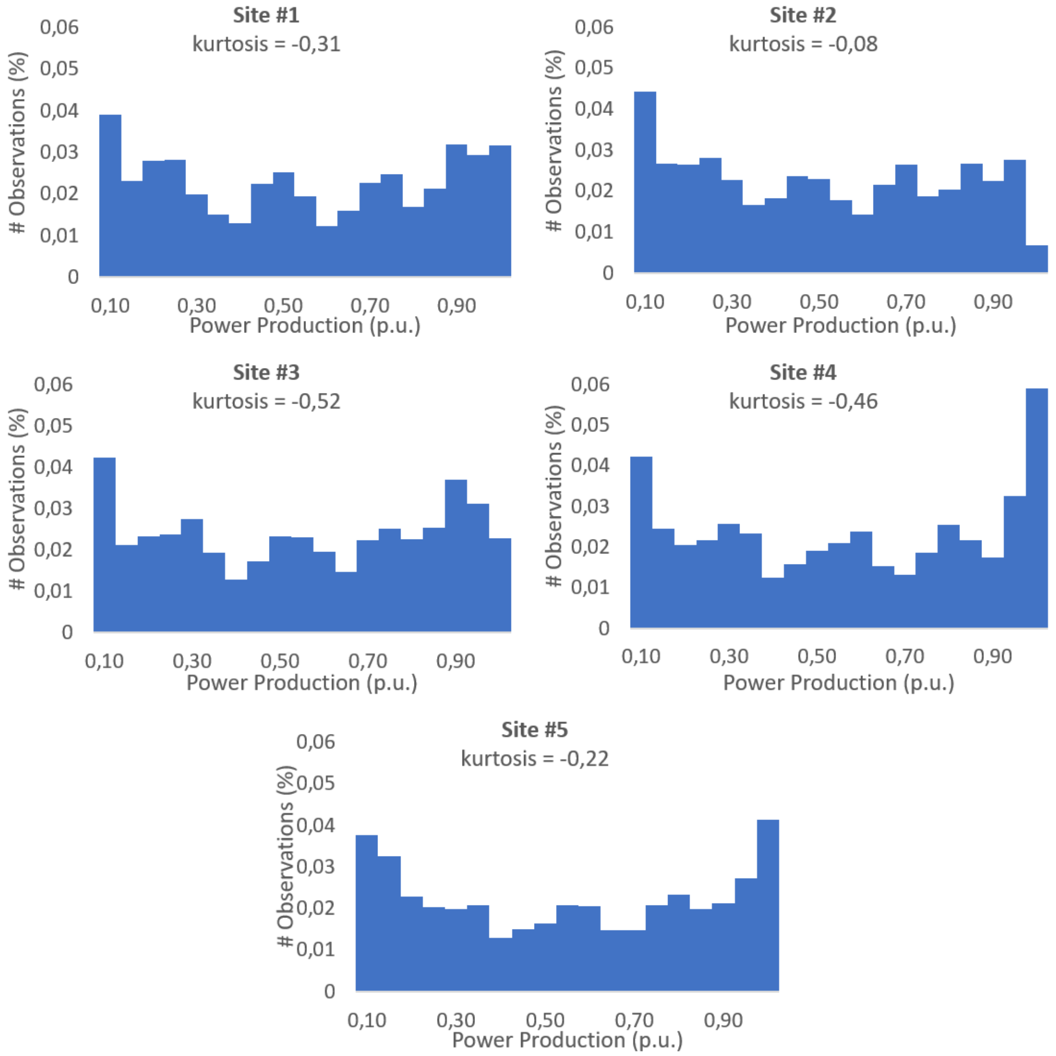


Figure 3.6: Histogram of the target variable - solar power production - in per unit (p.u.) values relative to nominal installed power.

From the comparison between sites, one of the standout results is site 2 where kurtosis is higher (although still negative, relatively closer to positive values), which seems to be caused by the lower weight of the right tail closer to nominal production capacity. This could suggest a site-specific condition, perhaps related to physical constraints on the installation of the panels (angles or shadows for instance) which makes it operate more frequently in a sub-optimal mode, from the perspective of using the maximum available production capacity.

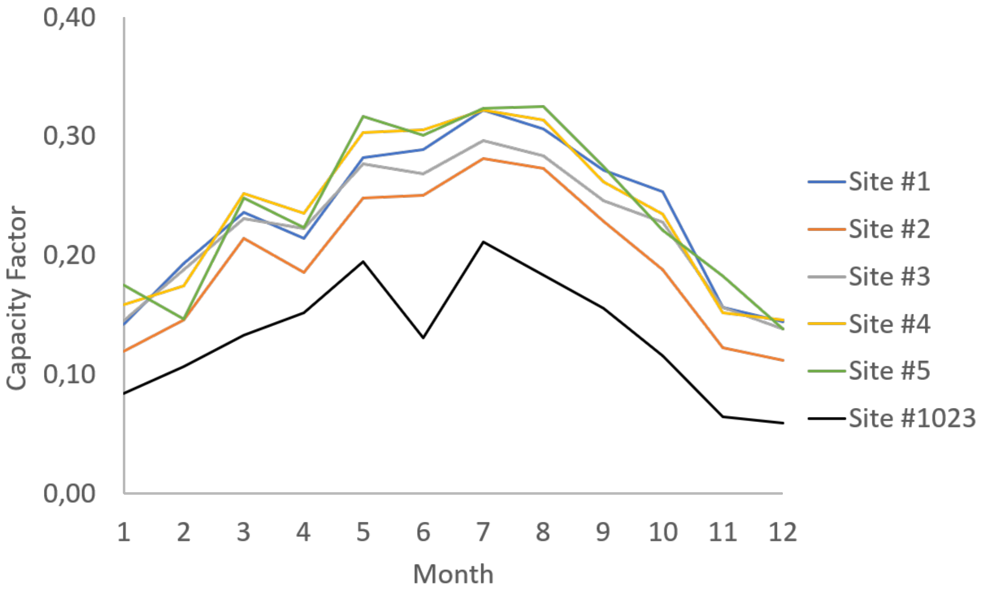


Figure 3.7: Monthly average capacity factor for each solar power plant.

Another interesting metric, particularly relevant in the context of solar power production, is the capacity factor plotted in monthly averages in figure 3.7. It corresponds to the ratio of energy produced by the system compared to an ideal production where the system would operate all the time at nominal power. The main difference between parks is related with site 1023 that shows a lower capacity factor for all the months when compared to other sites; this can be explained by it's location in the center while the other parks are in the south of the country. Additionally this site also exhibits a lower value in June caused by a period of production close to zero during June 2019 which is likely related with some downtime of the equipment. Then, the actual year pattern looks similar for all sites with a lower capacity factor during winter months and peak capacity during the summer, related with the seasonal patterns in the availability of the solar resource. In fact, during summer months, the capacity factor is typically 2-3 times higher for all the power plants, when compared to the winter period.

Bivariate Analysis

The core of the bivariate analysis will be the correlation matrix, depicted in appendices C, C.1 and figure 3.8, calculated only for the numerical variables using the appropriate Pearson correlation coefficient to measure the relationship between two variables. Although correlation is different of causality, this is still a useful tool to provide hindsight into how pairs of variables relate.

Looking into the correlation matrix, it's possible to find several pairs of variables exhibiting very high positive correlation above 80%. For instance, pairs of variables measuring the same physical variable at different atmosphere levels (*HGTlev1* and *HGTlev2*), temperatures (*temp* and *sst*) or wind speeds at different heights (*ulev1* and *ulev2*).

This type of information can be quite a valuable input to a correlation filter allowing to deploy some feature selection and enabling the use of a model with smaller number of variables without loosing critical data for the forecast. For instance, *u*, *ulev1*, *ulev2* and *ulev3* all contain information of the wind speed in the latitude direction, at different levels, presenting correlation values above 97%. This means that perhaps some of these are redundant variables, repeating the same information without bringing additional forecast insight to the machine learning model.

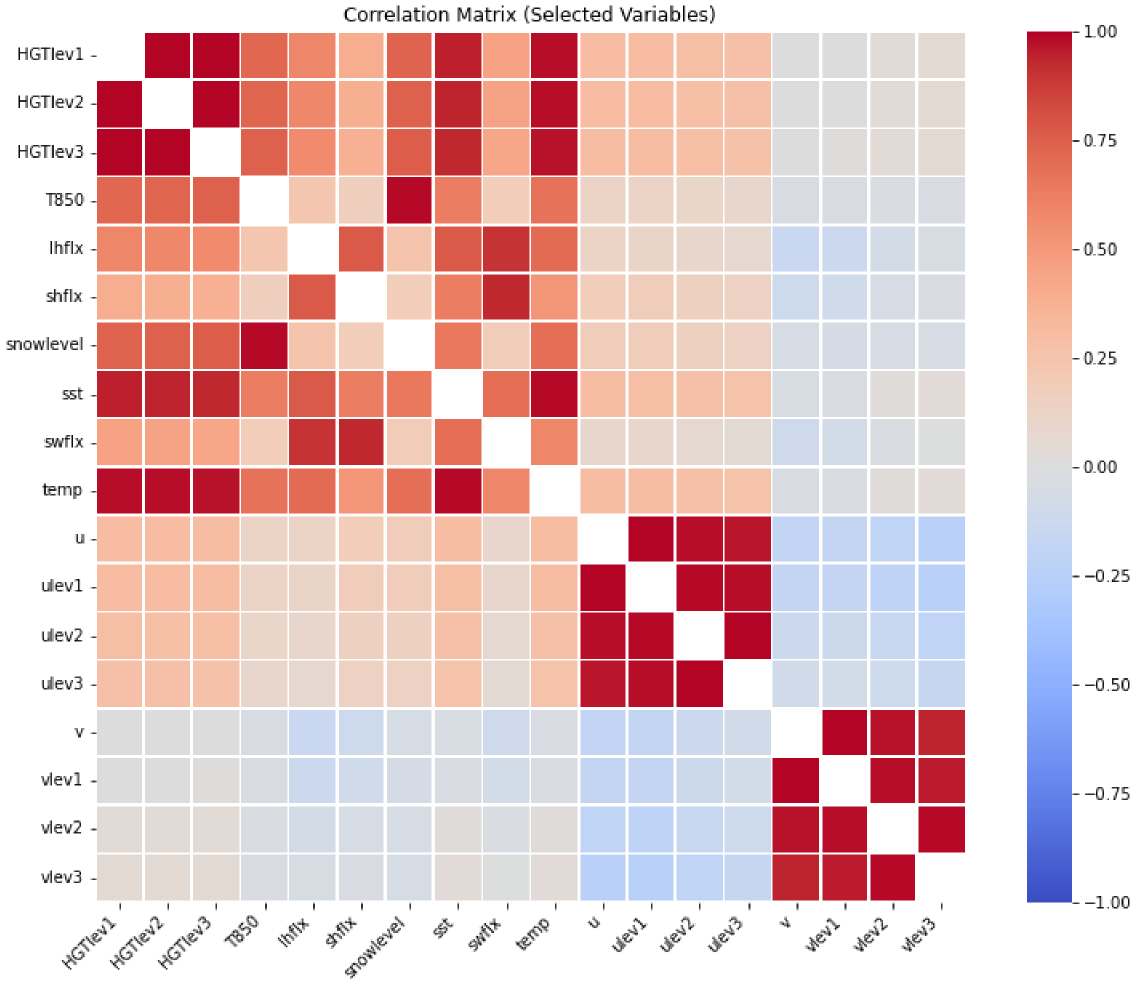


Figure 3.8: Variables with correlations above 90% in selected correlation matrix, calculated with data from site 5.

Target Variable

Besides the analysis between pairs of variables, it's also worth taking a detailed look into the correlation between the predictors and target variable, detailed in appendix C.2.

The analysis of these values, extracted from the correlation matrix, doesn't show any strong correlation, either positive or negative. Still, the highest correlation found (around 21%) happens with *swflx* variable which, as explained before, is likely to be a very good predictor due to it's nature; also validating what is suggested by its definition, long-wave flux *lwflx* is almost completely uncorrelated which makes sense since solar cells are not built to absorb that kind of radiation. Another interesting result is that the variables related with wind speed appear at the bottom of the correlation table.

3.3 Data Pre-Processing

The goal of this section is to present the methodologies used to deal with the raw dataset, built in Python as described in section 3.1, with the goal of enhancing the results of the forecast model. All these methods are implemented as options when running the Python script for the forecast. The complete list of options is presented, mostly for guidance, with a short description in table 3.3; the detailed description of each option is described in the following sections.

3.3.1 Feature Engineering

Feature Augmentation

Although the dataset contains a good amount of features to be used for solar power production forecast, this work also explores a technique commonly deployed in time-series related datasets. The base idea is to create cyclical time-related features, using sinus and co-sinus functions. A common example is that solar production in hour 23 should be more similar to solar production in hour 0 than solar power production in hours 10 and 20. However, using the hour as predictor, 0 and 23 are further away than 10 and 20. To deal with this problem, it's common to apply a transformation using these cyclical functions. The same rational can be used in the month where 12 (December) should be closer to 1 (January) than 6 (June).

This logic is applied both to hour and month, extracted from the timestamp variable as integers ranging from $[0, 23]$ and $[1, 12]$ respectively. Then, using both sinus and co-sinus, a total of 6 new time-related variables can be added to the original dataset: *hour*, *month*, *hour_cos*, *hour_sin*, *month_cos* and *month_sin*.

$$hour\_cos = \cos(\frac{2\pi * hour}{24}) \qquad hour\_sin = \sin(\frac{2\pi * hour}{24})$$

$$month\_cos = \cos(\frac{2\pi * month}{12}) \qquad month\_sin = \sin(\frac{2\pi * month}{12})$$

Finally, it should be noted that creating these artificial predictors can be quite useful if there's a need to make forecasts for future times where no meteorological predictors are available. Although far from ideal, it could provide a workaround solution is some specific contexts. The basic idea is that for any future timestamp, it's always possible to determine these time-related predictors and feed them into the machine learning model to produce forecasts even if there is no weather predictors available for such time horizons - such approach would constitute an alternative naive forecast model based on calendar variables.

Feature Selection

Although several machine learning based approaches exist to select the best features from a dataset, the basis approach of this work was to gather the knowledge obtained from the exploratory data analysis stage, presented in section 3.2, in order to make decisions on what variables to consider.

The first step was to drop the categorical variables which, as described before, are not relevant for solar power prediction. Then, also the variables that are constant over time were dropped since they have no discriminatory power, useful for the model.

The second step uses the knowledge from the correlation matrix to drop redundant variables containing similar information. This includes the following:

- Temperature measures such as *sst*, *T500* and *T850*. Only *temp* is kept as the temperature at 2m level, an height comparable with typical solar power installations.
- Wind speed measures in different heights and referential like *u*, *ulev1*, *ulev2*, *ulev3*, *v*, *vlev1*, *vlev2* and *vlev3*; wind direction *dir* is also dropped Only *mod* is kept, corresponding to the wind speed module at 10m height.
- Snow related variable *snow_level* is dropped since it is almost negligible for the sites under study.
- Geopotential height variables *HGT500*, *HGT850*, *HGTlev2* and *HGTlev3* are also dropped, keeping only *HGTlev1*.

Nevertheless, a more automatized feature selection method was also implemented based on tree boosting as described by Chen and Guestrin (2016). The XGBoost (extreme gradient boosting) algorithm can be used to determine the importance of each feature, with regards to the target variable, allowing to rank the predictors and providing a rational metric to discard features and simplify the dataset. This automated approached, fully based on machine learning algorithms, is compared (in the model construction phase) with the mixed approach described previously, where exploratory data analysis information is combined to support the decision on whether to keep or discard features. An example for this kind of approach is described in Huiting, Yuan, and Chen (2017) where the XGBoost algorithm is used to select features used for short-term electrical load forecast with the LSTM algorithm.

Using these automated and empirical methods, different setups were created to test different combinations of variables to feed into the model, in order to find the optimal set.

1. No Selection - keeping all the existing variables from Meteo Galicia except the ones with constant values which lead to issues in train stage.
2. No Selection and Feature Augmentation - all the Meteo Galicia variables plus the time predictors described in 3.3.1.
3. Custom Selection - variables selected following the rational explained in 3.3.1.

4. Custom Selection and Feature Augmentation - variables selected in 3.3.1 plus the time predictors.

5. Only Time Predictors - the base idea is to explore how the algorithm would perform if no meteorological data was available for some reason and it had to rely simply on timestamp information.

6. Only Radiation - using only *swflx* due to the characteristics evidenced and described in 3.5.

7. XGBoost Top10 - using the 10 variables with highest importance according to XGBoost algorithm.

8. XGBoost Top20 - using the 20 variables with highest importance according to XGBoost algorithm.

9. XGBoost Top10 + Time Predictors - using the 10 variables with highest importance according to XGBoost algorithm, selected from set including time predictors..

10. XGBoost Top20 + Time Predictors - using the 20 variables with highest importance according to XGBoost algorithm, selected from set including time predictors.

3.3.2 Outliers

For the target variable, the methods used to detect outlier observations were based on the specific physical assumptions:

- Filter for negative values: the electrical output of the photovoltaic system should always be positive meaning that power is output from the production system into the grid and not the other way around. Then, it makes sense to check the target variable for negative observations. The handling method is quite basic and negative values are set to zero instead.

- Filter for installed power: any electrical production system is characterized by its nominal or installed power, i.e., the maximum amount of power it can output. It's then appropriate to check for values above, considering a default tolerance of 5% since it's also typical the real power to be slightly above the nominal mentioned in the equipment data sheet. The flagged observations can be handled in two different ways: use linear interpolation between the previous and next observation to obtain an estimated value or simply set the observation to the nominal power of the equipment. The later option is mostly relevant for the post-processing stage as the model doesn't include constraints to the nominal power of the equipment which means it may output predictions above that threshold.

- Filter for solar time: Empirically, there should be no solar power production when there's no sun. To implement this sanity check, a filter was used that takes the site location and uses Python *astral* package in order to determine the hours of sunrise and sunset for all days in the dataset. Then, two basic conditions apply: if sun hasn't risen yet or if sun has set already, production cannot be more than zero; if otherwise, the observation value is set to zero.

3.3.3 Missing Values

Although Keras model documentation refers that, in general, missing values can be replaced with zeros for the purpose of being used by neural networks, that is not the case with this dataset because zero actually has a meaningful value representing lack of power production. Several methods like replacing with mean or other statistical measure or using some kind interpolation can be applied. For this work, and since missing values correspond only to a small amount of data, a more simplistic approach was used and any observation (row in the dataset) containing missing values was dropped.

3.3.4 Scaling

References such as Chollet (2021) warn that it's not a good practice to feed data that takes large absolute values into a neural network or data that is quite heterogeneous with variables in narrow ranges and others in very large ranges. The dataset, described in previous sections, clearly fulfills both conditions specially due to the different physical units used in the underlying data. The way to solve this problem and facilitate the learning task was basically to apply strict normalization to the data with two possible options: normalize based on minimum and maximum value, for each variable, with resulting data comprised in $[0, 1]$; normalize based on mean and standard deviation, for each variable, with resulting data comprised in $[-1, 1]$.

Problem			**Option**	**Description**
NA Handling			none	Keep observations containing missing values.
			remove_obs	Remove observations containing missing values.
Scaling			none	Keep original scaling of variables.
			minmax	Normalize data into [0,1] range.
			normal	Normalize data into [-1,1] range.
Feature Engineering	**Augmentation**		none	Don't add new features.
			time_predictors	Add 6 variables related to hour, month and their sin/cos transformations.
	Selection		none	Don't remove any features.
			xgb_top10	Top 10 variables selected by XBoost according to feature importance.
			xgb_top20	Top 20 variables selected by XBoost according to feature importance.
			meteo_galicia	All meteo variables except categorical and constants.
			custom	Selection of meteo variables based on 3.3.1.
			only_swflx	Keep only swflx radiation feature.
			only_time_pred	Use only the augmented time-related features.
Outliers	**Target**	**Negative Values**	none	Keep negative values.
			zero	Replace negative values with zero.
		Power	none	Don't check against installed power.
			linear_interp	Replace values with interpolation between value before and after.
			set_to _max_power	Replace values with maximum installed power.
		Solar Time	none	Don't check against solar hour.
			sunrise_sunset	Replace production out of sunrise-sunset with zero.

Table 3.3: Summary of all pre-processing options explored.

Chapter 4

Forecast Model

4.1 Forecast Model without Transfer Learning

4.1.1 Test Setup

The data from Meteo Galicia weather service comes in at different time horizons from D-3 (data for three days ahead of present time) to D0 (data for the current day). Handling these different time horizons is the key to create a robust and flexible forecast model which can be deployed for real time application. Consequently, some decisions are done when creating the different datasets to use in the traditional machine learning workflow of train, validation and test.

The machine learning models are trained with data from the past describing both predictors and target variable. Thus, it makes sense to use past weather data from horizon D0: standing in present time and looking back into the past, all horizons should be available so it makes sense to use the weather values as close to the forecast day as possible instead of predictions made 2 or 3 days before.

Then, standing in the present time, there's a forecast of weather variables for the next three days, so it makes sense to use a forecast window of the same size which will be called real time forecast, using as predictor the D-3, D-2 and D-1 horizons from Meteo Galicia. On the other hand, trying to emulate the market daily nature, it also makes sense to use D-1 data for the predictors in a situation comparable to use the day-ahead weather forecast to define the day-ahead energy acquisition in the market. The two options can be summarized as below.

- **Production setup:** can be used in a real-world situation where Meteo Galicia data is available only for the following three days. The forecast window will then have three days with D-1, D-2 and D-3 predictors for each one; the test set will then be composed of the hourly predictors for 3-days ahead of present time. From a production point of view, for the same day, three different forecasts will be produced based first on D-3 predictors, then D-2 and, finally, on the day before, with D-1 weather predictors. This corresponds to an update of forecasts for the same day, as better weather information becomes available; the logic is depicted in 4.1.
- **Development setup:** based only on D-1 Meteo Galicia data, this setup cannot be used

for real world deployment because only one D-1 day is available. Nevertheless, it is a useful tool for model development making predictions for the past. From a production point of view, this could be adapted as the equivalent of making only next day forecast (based on D-1) for each day thus never using D-2 and D-3 predictors. For the purpose of this work, a 10-day window was implemented as a good compromise between having enough forecast days to evaluate the algorithm and still keep a manageable run time for the algorithm.

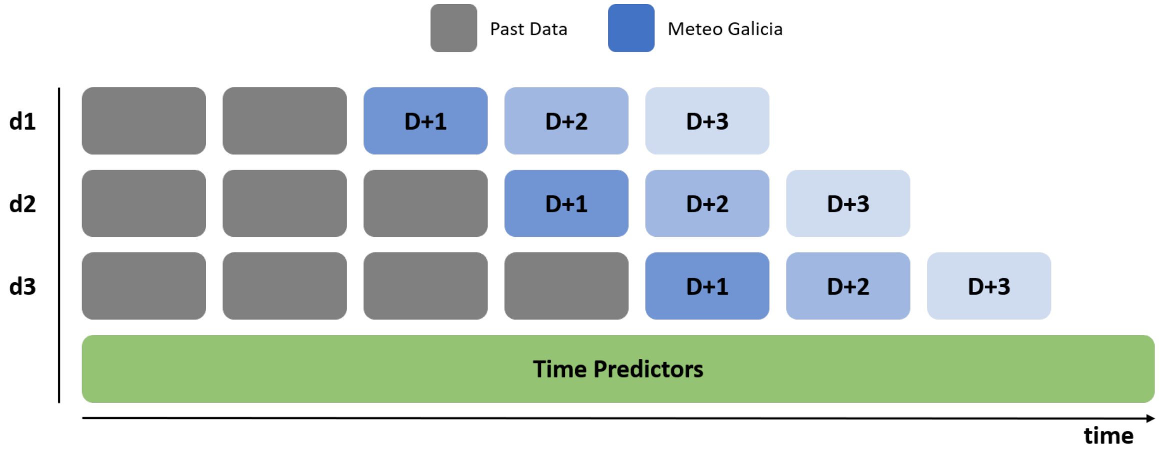


Figure 4.1: Available data for forecast in each time step, highlighting how predictions can be replaced as more updated meteorological data is collected.

For both setups, the train set in automatically defined as all the data available at the time of the forecast; from this train set, a certain amount (10% or 20% are considered in this work) is kept for validation. A notable detail, characteristic of of time-series data, is that these data splits to generate train, validation and test sets must take into account the arrow of time; unlike other types of data where random splits and sampling can be used to generate these sets, on time bounded data this cannot be done, otherwise there's a risk of training the model with data from the future when the idea must be to predict the future by looking into the past. So, from the past into the future, the train set is the first partition, then the validation and only afterwards should come the test set. The procedure is schematized and exemplified in figure 4.2.

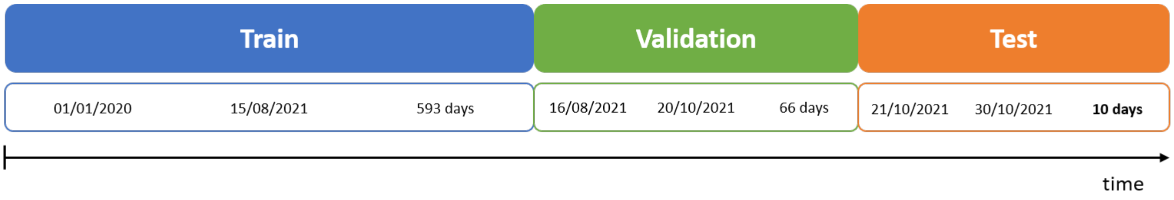


Figure 4.2: Timeline of train, validation and test splits with an example using 10% validation ratio in the development setup.

4.1.2 Model Design

The basic deep learning network used in this work is based on LSTM cells that allow to keep information across different time steps effectively saving information for different time steps and avoiding that older information is forgotten. The final choice for LSTM as the main model for this work is justified by the results presented in section 5.1.1; nevertheless, to reach this final decision, different models were also implemented and tested:

- **H-1:** A naive baseline model where the forecast value for any chosen hour consists simply in the previous hour observation for production. This is not a real world possibility because the production data has a delay to become available and also the market opening and closing hours would not allow to have such an operation. Nevertheless, due to its simplicity, is an interesting point of comparison for the deep learning model.

- **Regression Tree:** A regression tree model based on the *sklearn* library, described by Pedregosa et al. (2011), was also used as another benchmark model. This kind of model is a decision tree that outputs continuous values thus being suited for regression tasks. The default model parameters are used for and no special hyperparameter optimization was implemented in its use.

- **RNN:** A simpler recurrent neural network (RNN) was also implemented through the Keras library; this kind of model are usually well suited to time-series or language models. When compared to LSTM, a basic RNN is typically less capable to retain longer patterns due to the vanishing gradient problem.

The complete process followed to design the machine learning model and output the power production forecast is described in the following sections and illustrated in figure 4.5.

LSTM Model

The key aspect that differentiates the LSTM layer from a simple recurrent layer relates precisely with the vanishing gradient problem, an effect characterized by the fact that the network becomes untrainable as layers are added to build the deep learning network. So, this feature allows to stack several layers in sequence, using Keras sequential model, thus enhancing the representational power of the network and getting a better forecast error. In this kind of design it's essential that all the intermediate layers return the full sequence of outputs, to be used by the following layer, as explained by Chollet (2021).

Although using stacked LSTM layers can typically provide good results, it's often a good idea to deploy also some kind of regularization technique to avoid getting the network training to fall into an overfitting situation. Dropout is one of the simplest methods for regularization in a neural network; basically, some neurons are ignored during training in order to avoid that the network will adapt too much into specific examples thus allowing to have a better generalization to decrease the error in the test set. This can be easily done in the context of a Keras sequential model using Dropout layers to connect LSTM layers. These layers are defined basically by the *rate* argument where the percentage of input units to be dropped from training is defined. Another way to avoid this issue, after the basic layers for the model are chosen, is to reduce the size of the model which is determined both by the number of layers and the number of input units by layer; this will be discussed later in section 4.1.2.

The last (hidden) LSTM layer will then be linked to the output layer where the neural network model presents the prediction; in this particular case, that will be a normalized prediction that will need to be re-scaled, an issue that will be addressed in section 4.1.4. The network topology presented is then a stack of layers which will map the inputs into a single output, represented by a dense fully-connected layers with 1D dimension which is appropriate in the context of time series forecast.

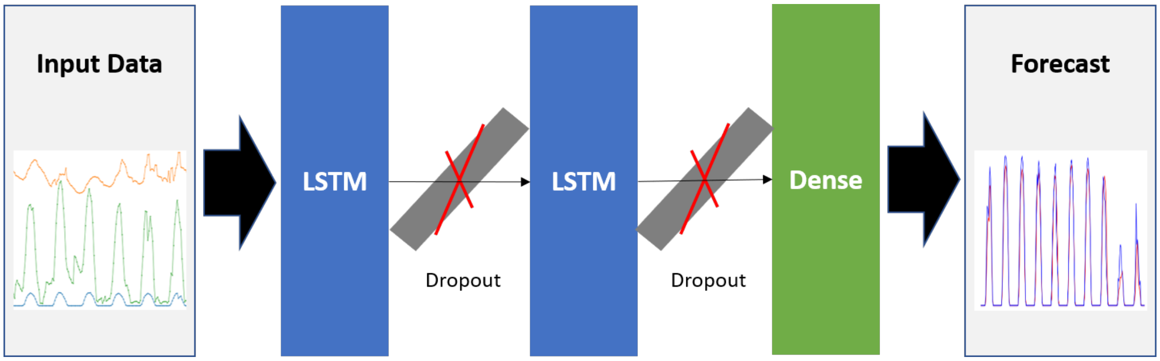


Figure 4.3: Basic structure of Keras sequential model with LSTM and dropout layers.

Tensor Structure

One of the basic building blocks of the deep neural network model used in this work is a long short-term memory (LSTM) layer, a type of recurrent network. For the Keras model implementation, it's necessary to rearrange the dataset into 3D tensors to match the network input that consists of three dimensions: samples (each consisting of one sequence), time steps (point in time where observation is taken) and features (one observation at a specific time step).

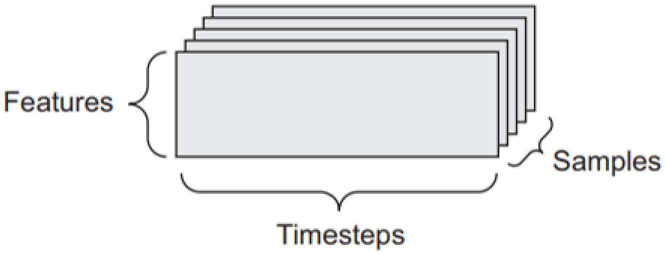


Figure 4.4: Structure of tensor for time series data, depicted in Chollet (2021).

Hypermodel Tuning

It was mentioned before that overfitting is a characteristic problem prevalent in deep learning models which are can typically grow into very large size models. Although default Keras parameters and layer configuration can achieve good results, this work also explored the use of Keras tuner to perform the hyperparameter optimization through some chosen search algorithm.

First, it was necessary to create a function that delivers instances of an hyper-model where its parameters can vary within controlled ranges. The basic idea is to find the optimal number and dimensionality of the layers but other parameters can also be added into the search space. Starting with a LSTM layer and then alternate Dropout and LSTM layers; the maximum number of allowed LSTM layers is an input set to 5. Additionally, two different hyperparameters can be explored when creating an instance of the hyper-model: the number of inputs units of each LSTM layer (allowed to be set from 32 to 512 in steps of 32) and the dropout rate of each Dropout layer (a random value from 10% to 50%, not too low to have some effect and not too high to avoid under learning). Then, a final 1D dense output layer is also connected for the reasons explained in 4.1.2. Finally, the model is compiled with the learning rate (allowed to be chosen from a set of pre-defined values), setting the optimizer (Adam algorithm, a stochastic gradient descent method, suited to large models, is used by default) and the loss function. For the later, lot of different options exist to calculate the value that the train process seeks to minimize; in this case, a measure of the model fit towards validation data. In this work, MAE (mean average error) was used.

The Keras tuner framework, described by O'Malley et al. (2019), allows to have an effective search through different models to find the best topology. With the search space defined in the hyper-model function, an objective is defined to chose between models. In this case, the tuner has been set to use the model loss function for that purpose. Then, two parameters are set to define how long and how hard to search for: the number of trials to run during the search and the number of models that are built and fit in each trial. Within the same trial, the hyperparameters are the same and the idea is simply to reduce the results variance; for each trial, new hyperparameter values are generated.

With the hyperparameters search space and trials defined, the last step is to define the search algorithm to be used. The base tuner class can be instantiated to correspond to different algorithms that manage the build, train, evaluation and save of each Keras model version. For the purpose of this work, a random search tuner was primarily deployed with some experiments also conducted using the Bayesian optimization tuner with Gaussian process; other tuners such as Sklearn (for Scikit-learn models) and HyperBand algorithm are also available but were not used in the context of this work. Additionally, Keras also provides a *get_best_hyperparameters* built-in method that can be deployed to get the best untrained model found during the search process conducted with the tuner.

4.1.3 Train and Validation

After the deep learning network model topology is defined and the hyperparameters optimized, the model should be ready to be trained on the data. This section will explain the standard process where data for a single site, split in train, validation and test sets, will be used to train the model and then make predictions for that same site; more complex setups, enabling transfer learning, will be discussed in 4.2.

The model is trained using the train data based on the splits defined in 4.1.1 with 10% of the train data taken apart for validation, meaning that it's not used for actual training. Then, the model fitting is performed defining a maximum number of epochs (iterations over the entire set of data) to avoid the process being trapped in very long train cycles. The validation set is also provided as an argument: although the model is not actually trained on it, it will be used to evaluate the loss function at the end of each epoch. For the train process, two different callback functions are used to manage the model during training, effectively guiding the process and providing options for posterior use.

First, the early stopping callback allows to stop the training before the maximum number of epochs is reached, based on the monitoring of some metric which stops improving. This process is also helpful to avoid overfitting to the train data which should theoretically improve the generalization towards the test set. This callback function is defined by the metric to be monitored, by the number of epochs without improvements after which the training should be stopped and a minimum value of loss function improvement to be qualified as improvement.

Second, the model checkpoint callback is used in order to save the model or its weights. The setup configured for this work only saves the model weights in the epoch where it is considered to be the best according to monitored loss function, thus guiding the problem objective of achieving the minimization of that measure.

Finally, when the train process stops, it's possible to plot the evolution of the loss measure, both in the train and validation sets, effectively providing a visual insight into the results of the model fitting.

4.1.4 Forecast

After the model is trained using the process described in 4.1.3, it can be used to make predictions for the test set described in 4.1.1. As explained before, scaling is a fundamental step for the Keras deep learning model and the raw predictions will correspond to scaled values. Thus, these raw predictions must be scaled back using the appropriate method depending on how data was normalized in the pre-processing stage.

Post-processing

After the predictions are scaled back to MW or MWh units, the forecast model has achieved its goal. Nevertheless, it's also important to keep in mind that the same considerations made in 3.3.2 should still apply to the predictions because they are bounded by the same physical constraints of the photovoltaic production system. This is brought into play in a post-processing stage of the forecast workflow.

In practical terms, this means that forecast production values, $\widehat{y}$, will be further constrained by the following conditions:

- Production bounded by solar hour

  $if \quad time \quad \epsilon \quad night \quad then \quad \widehat{y} = 0$

- Production cannot take negative values:

  $if \quad \widehat{y} < 0 \quad then \quad \widehat{y} = 0$

- Producation cannot exceed system capacity - nominal power (P_{NOM}):

  $if \quad \hat{y} > P_{NOM} \quad then \quad \hat{y} = P_{NOM}$

Error Evaluation

To evaluate the performance of the forecast model, the predicted values can be compared to the actual values corresponding to the test. To measure the distance between forecast and actual values, three different metrics are considered:

- Mean Absolute Error (MAE) - also generally used as the loss function while training the algorithm. Simpler to interpret as the average value of all the errors.
- Root Mean Square Error (RMSE) - useful for comparison since it penalizes larger errors, although more sensitive to outlier values which shouldn't be a problem after post-processing.
- Coefficient of Determination (R^2) - convenient for comparison across models since it's scaled between 0 and 1.

Although these metrics can provide relevant insight into how the model performs across sites (R^2) or for different test setups varying with different amount of historical data, they still mean little into why the LSTM-based model should be chosen. So, other forecast models where considered for benchmark.

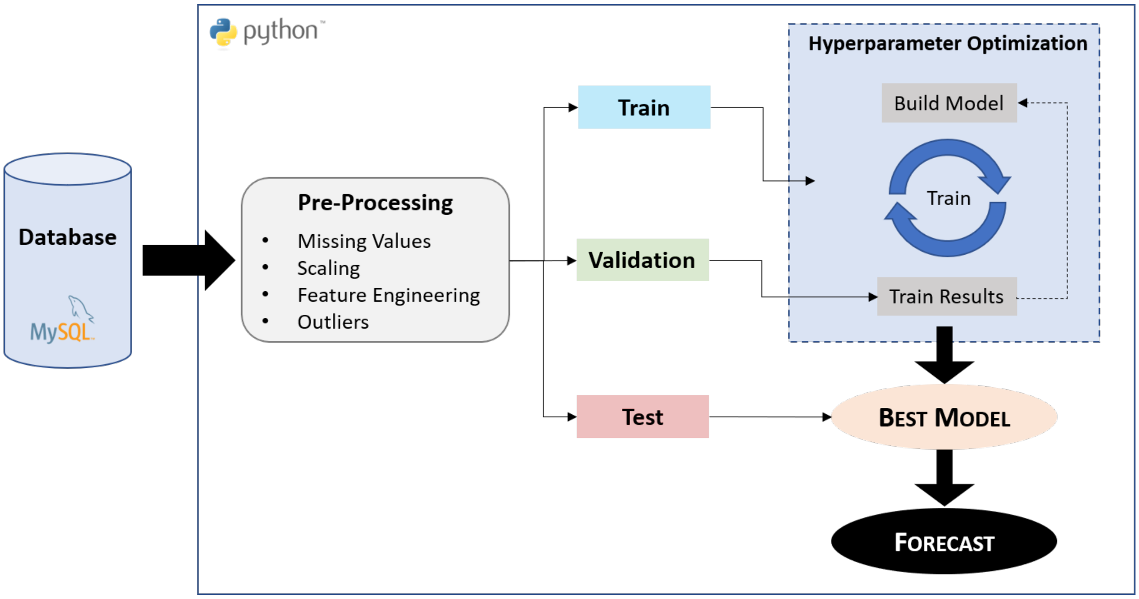


Figure 4.5: Overall view of pre-processing, optimization, training and forecast tasks in the forecast model workflow.

4.2 Forecast Model with Transfer Learning

The baseline for this work was to create a forecast model for solar power production based on historical data which can be quite scarce on new photovoltaic projects, thus fitting the characteristics of the cold-start problem. Then, in this section, the work explores the possibility of applying the transfer learning paradigm to improve the forecast performance on such conditions.

From all the transfer learning approaches described in chapter 2, this work explores transductive transfer learning since the source and target tasks are the same (solar power production forecast) and the source and target domains are different yet related. The approach in this work can also be classified as parameter-based in the sense that the means of knowledge transfer will be the weights of pre-trained models. The source models can be used either for feature extraction (freezing all layers except the final ones) or weight initialization (all the layers are fine tuned with target data). The second approach was finally dropped both due to the excessive sensibility that it showed towards the selected target data used and also because feature extraction mode actually provided better error values as can be verified in chapter 5.

4.2.1 Test Setup

Regarding the process and code implementation to enable transfer learning, different steps are taken. One of the key requirements, fulfilled during the implementation of the algorithm, was to have the possibility of choosing any possible combination of source sites in order to build a forecast for any selected target, from the available sites. This flexibility enables the test of different hypothesis to investigate the potential of transfer learning in different scenarios.

The first step is to define which source sites will be used to train the model; more specifically, after selecting the target site, how to choose the source sites that can be of greater value for the prediction, minimizing the forecast error. The selection of the most suitable source domains (or production sites) is based on the idea of similarity leading to the transferability of knowledge between sites sharing common features. Although more sophisticated approaches can be used, in this work two simple methods were used to guess the best source sites for each target: the straight line distance between sites and the coefficient of correlation of their power production time series.

- the geographical distance between the target and source sites, defined by their latitude and longitude, under the assumption that if they are closer, then power production should also follow similar patterns. It should be highlighted that such an approach disregards unavailable site-specific information such as the shadow patterns or the solar panels azimuth and orientation. The calculation was made using the *geopy* Python package.

- the Pearson correlation coefficient calculated for the solar power production time-series between different pairs of sites. The calculation was made using the *corr* method available for *pandas* DataFrame structures in Python.

Then, the number of source sites to consider could also be a subject for further studies. In this work, where five sites are closer together in the south of the country and another one further up north, using two source sites has been found to be a balanced compromise between achieving a good forecast result and keeping train time at a reasonable amount.

The final step is the decision on the range of data to consider. This work explores the possibility of taking in different amounts of data from source and target sites; for an extreme case, it can even be considered that no data exists for the target site, a situation where the forecast would be effectively produced based only on a model trained with source sites. The most interesting outputs expected from this test setup are the evaluation of transfer learning effectiveness and insight into how much data needs to be collected for the target site in order to outperform the transfer learning model.

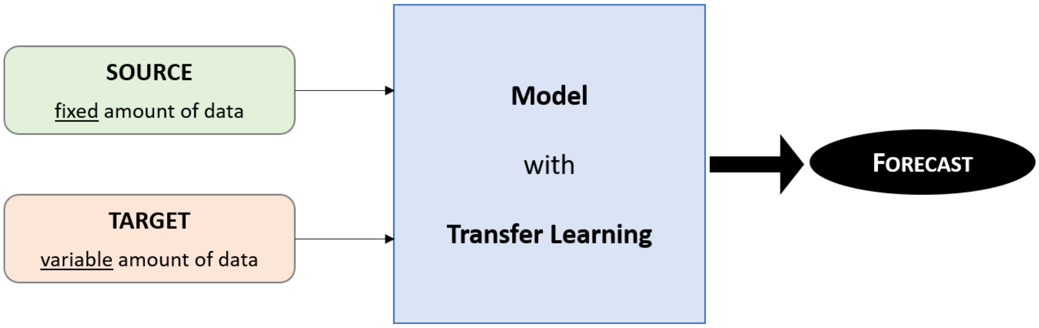


Figure 4.6: Schematic of the strategy used to explored the impact of the size of target data in the forecast.

4.2.2 Train and Validation

The implementation of transfer learning is based on the processes described in 4 but with a few subtleties. The main sequence consists of reading the data from first source site, perform the defined pre-processing options and create the splits in dataset. Here lays the first major distinction in the TL from the normal process: while before, the split consisted on train, validation and test, when dealing with source sites no test is necessary because no forecast will be produced. The split can be then limited to train and validation. The model design, if based on hyperparameter optimization, is done only when the first source site is processed. Afterwards, the model design is considered to be locked and only the weights of the network can be tuned on the next stages. The process continues with the second source site with the pre-processing and split into train and validation; the weights of the model are then updated with the new data; this procedures is repeated for all the specified source sites.

When all source sites are processed and the model trained with their data, the transfer learning continues with the target site data. Here, after the pre-processing, the data split will now include the typical train-validation-test split because the goal is now not only to train a model but also to actually produce a forecast. Before training with target data, it necessary to set the trainable status of each layer: by default, all layers can be trained; for the purpose of transfer learning, particularly when used in the feature extraction variant, some layers are frozen which means that training with target data will not affect them. Then, the training process goes on normally. It should be mentioned that both the feature extraction and weight initialization approaches of transfer learning, schematized in figure 4.7, will be explored in section 5.2.2.

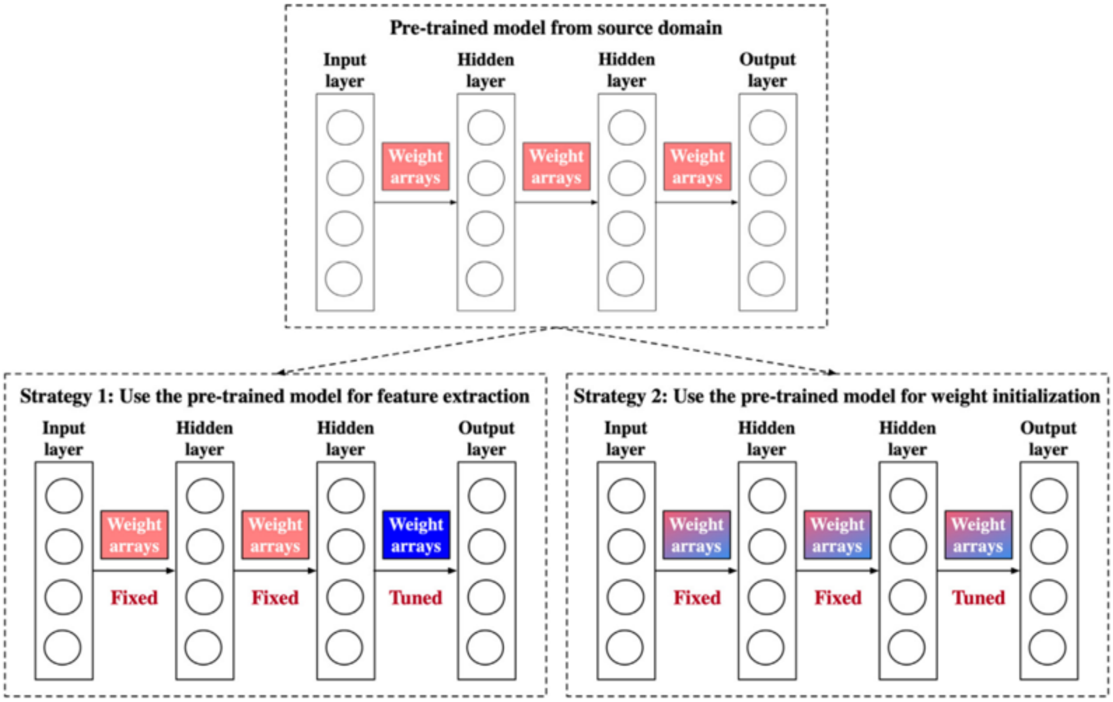


Figure 4.7: Deployment schemes of network-based transfer learning from Fan et al. (2020).

4.2.3 Forecast

Through all these stages of reading data for each site, either source or target, the model is saved. Starting on the first source site, after the train and validation stage, the model will be considered to be trained on that data and saved. Then, the data from the second source site will be used to further train the model which was previously saved after the first source site. The process is repeated until all the source sites are used to train the model. Afterwards, a similar process occurs with target site data, used to run the final training of the model which is will be then reflecting the train effects both by source and target sites. It can be then used to produce a forecast for the test set. The process will finalize with the evaluation of the forecast error, not just for the final model (trained with source and target data) but also for the intermediate models that are saved throughout the procedure. Such setup allows to assess the potential and effectiveness of transfer learning through all the algorithm stages.

The figure 4.8 presents an example where the goal is to forecast for site 5 using the data from source sites 1 and 2. The setup allows to evaluate how accurate is the forecast when the model is trained on three different stages: only data from 1, data from 1 and 2 (sources) and on data from 1, 2 and 5 (sources and target).

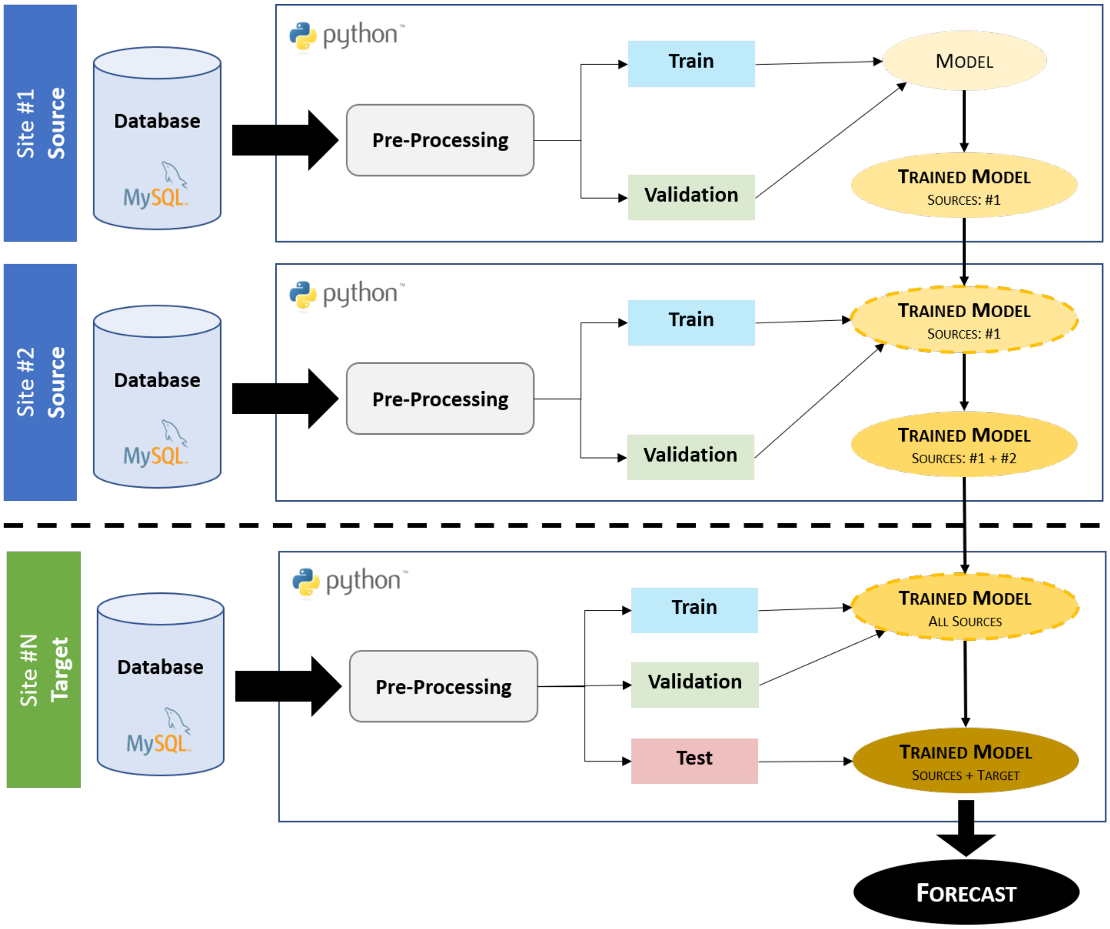


Figure 4.8: Overall view of pre-processing, optimization, training and forecast tasks in the model with transfer learning.

Chapter 5

Results

5.1 LSTM Model

5.1.1 Benchmark

The first major result was simply to verify how the LSTM model could compare with other models described in 4.1.2. For that, the development setup defined in section 4.1.1 with 10 days of forecast, was used with a LSTM model without hyperparameter optimization, using 1 year data from site 1 and D-1 predictors. Both pre and post-processing options were equal for all models in order to have comparable results.

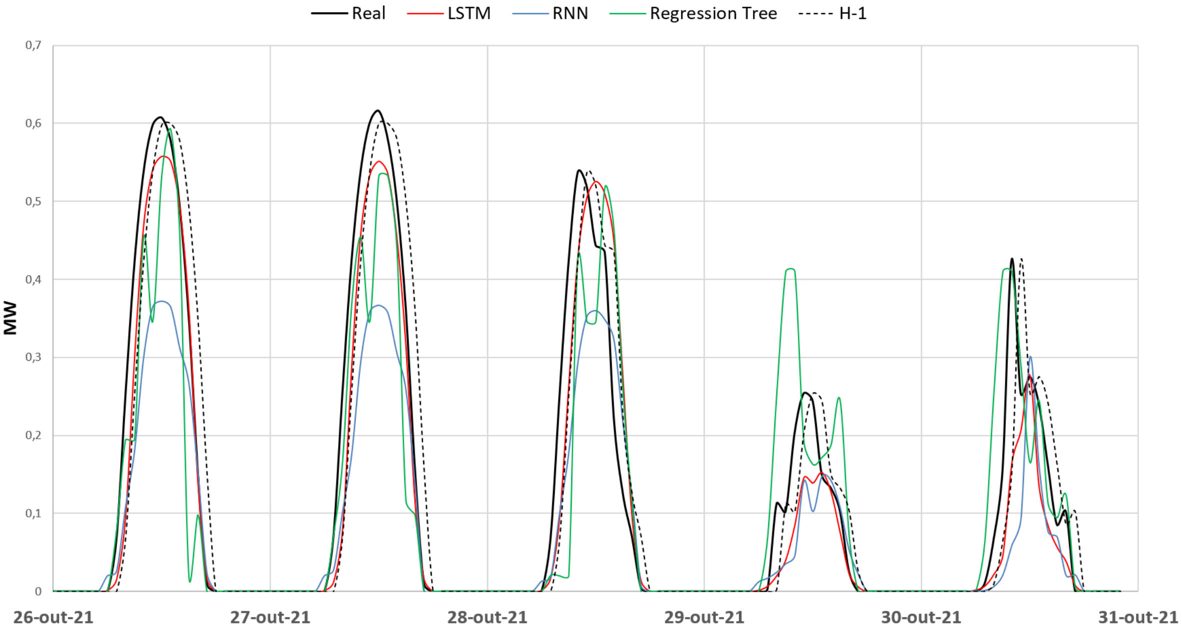


Figure 5.1: Final 5 days of 10 days forecast, using the development setup for site 1, with different machine learning models for benchmark.

Site	Best Model		Next Best Model		Δ RMSE
	Type	RMSE	Type	RMSE	
1	LSTM	0,064	H-1	0,082	22%
2	LSTM	0,740	Regression Tree	1,092	32%
3	LSTM	3,220	H-1	5,586	42%
4	LSTM	0,396	H-1	0,599	34%
5	LSTM	2,721	Regression Tree	4,252	36%
1023	H-1	2,853	Simple RNN	3,911	27%

Table 5.1: RMSE error of the best and next-best models for each available site.

The results presented in table 5.1 clearly indicate that LSTM model outperforms all the other models on the RMSE metric for all sites with the exception of 1023; this particular result probably originates from the fact that the production series for site 1023 contains periods with no production related with some downtime of unknown cause. For the remaining sites, the best forecast results are obtained with LSTM-based model which outperforms the second best model (the previous hour model H-1 or regression trees) by 20% to 40% on the RMSE error, depending on the site. Such outcome backs the decision to invest on tailoring and refining the LSTM model because it seems to have the biggest potential for accurate power production forecast.

5.1.2 Impact of Hyperparameter Tuning

The next experiment was aimed at exploring different variation for the network. This was done using the *Keras* built-in tuner by allowing the search through different configurations for the best one as evaluated by the loss function MAE. The setup consisted of 5 different iterations of the algorithm, for a specific site, where the only differences are the ones randomly introduced by the Random Search algorithm used in the tuner.

Interestingly, the results presented in table 5.2, show that the smaller model with only two layers actually performs better on all metrics when compared to larger models with extra layers. This could be an hint that even with the early stop callback in place, the model may still be suffering from overfitting towards the train and validation set and then failing to generalize as well to get a better performance on the test set.

Impact of Search Algorithms

The Keras framework allows the use of different search algorithms to find the best model configuration based on the defined hyperparameters. An experiment was devised where five iterations were run for the same site using two different tuners based on Random Search (RS) and Bayesian Optimization (BO) algorithms.

The results, ordered by RMSE in figure 5.2, show that the best performance was obtained with a model configuration built with RS algorithm. Looking at all iterations, there's no clear pattern that allows to state conclusively that either algorithm is the best. Moving forward, the decision was to take the RS algorithm as the simplest implementation of a Keras tuner.

Iteration #1		Iteration #2		Iteration #3		Iteration #4		Iteration #5	
Layer	**Units**	**Layer**	**Units**	**Layer**	**Units**	**Layer**	**Units**	**Layer**	**Units**
LSTM	352	LSTM	96	LSTM	192	LSTM	224	LSTM	256
Dropout	352	Dropout	96	Dropout	192	Dropout	224	Dropout	256
LSTM	32	LSTM	320	LSTM	224	LSTM	32	LSTM	32
Dropout	32	Dropout	320	Dropout	224	Dropout	32	Dropout	32
LSTM	32	Dense	1	LSTM	32	LSTM	32	LSTM	32
Dropout	32			Dropout	32	Dropout	32	Dropout	32
LSTM	32			LSTM	32	Dense	1	LSTM	32
Dropout	32			Dropout	32			Dropout	32
Dense	1			Dense	1			Dense	1
Metric	**Value**	**Metric**	**Value**	**Metric**	**Value**	**Metric**	**Value**	**Metric**	**Value**
MAE	0,0341	MAE	0,0258	MAE	0,0317	MAE	0,0465	MAE	0,0336
RMSE	0,0609	RMSE	0,0483	RMSE	0,0582	RMSE	0,0801	RMSE	0,0603
R^2	0,9379	R^2	0,9516	R^2	0,9419	R^2	0,9348	R^2	0,9404

Table 5.2: Performance metrics and network configurations obtained in different iterations of hyperparameter optimization, for site 1.

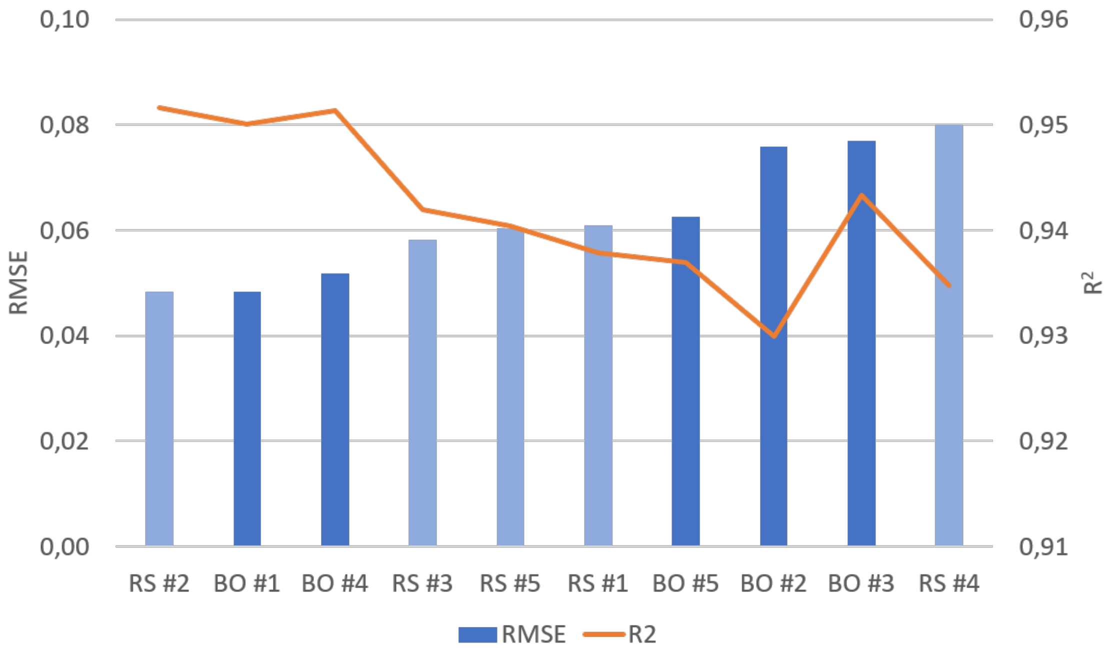


Figure 5.2: Performance metrics for multiple iterations of hyperparameter optimization, using different Keras tuners, for site 1.

5.1.3 Impact of Feature Engineering

Another interesting study is the sensitivity of the model towards different combinations of predictor features described in section 3.3.1. This was done simply by taking different sets of features running on the same model configuration, pre and post-processing setups. The model configuration with best performance in 5.1.2 was selected for this test.

Setup	**MAE**	**RMSE**	**R^2**
(4) Custom + Time Predictors	0,0285	0,0510	0,954
(10) XGBoost Top20 + Time Predictors	0,0340	0,0594	0,949
(2) Meteo Galicia + Time Predictors	0,0364	0,0638	0,948
(5) Time Predictors	0,0434	0,0809	0,851
(9) XGBoost Top10 + Time Predictors	0,0442	0,0811	0,908
(8) XGBoost Top20	0,0501	0,0908	0,902
(1) Meteo Galicia	0,0523	0,0918	0,915
(3) Custom	0,0576	0,1001	0,920
(6) Radiation *swflx*	0,0617	0,1092	0,833
(10) XGBoost Top10	0,0620	0,1099	0,898

Table 5.3: Impact in performance of different feature engineering options, ordered by RMSE.

The results show that the custom feature selection method effectively translates into an improvement in the forecast error metrics; in fact, it makes the RMSE improve by roughly 20% when compared with the case where all the Meteo Galicia variables (except constant values) are feed into the model. Such results suggests that LSTM algorithm benefits by selecting the best features from data and use them to make a better forecast.

Another notable result is that the feature augmentation method used in this work, although quite simple, provides a notable boost in the forecast quality. In fact, when comparing the Custom setup, with and without the augmented features, a improvement of almost 50% is found in RMSE when the time features are added into the dataset.

Then, it's also quite interesting to find that the dataset set using only time predictors and no meteorological data have a performance not much worse than the dataset that takes only the meteorological data. Again, this fact underlines the power of feature augmentation and also the possibility that in real world applications, if any issue arise with the reception of meteo data, within the appropriate time frame to send forecasts to market, a reasonable backup solution can be found just by working with these timestamp related predictors.

It was also proved that while radiation and solar power production share a relation of causality, with the first leading to the second, it's not the best solution to take as predictor. One of the reasons could be because the radiation is measured on a horizontal plane which is different from the typical angle of solar panels.

Regarding the automatic feature selection method, based on XGBoost, the variable rank contained in appendix D provides some interesting insights. The two most important variables are related with geopotential height, the height above sea level of some atmospheric pressure; one possible explanation could be that atmospheric pressure is actually a proxy to the weather: lower pressures typically correspond to cloudier weather and higher pressures to cleaner sky. Then come hour and month, two features originated from the augmentation process thus underlying its relevance. The radiation related variables are also part of the top 10 together with temperature which again may be a proxy of the general weather state. The best setup including this automated selection consists in taking the top 20 variables, selected from a set including the time predictors, and it provides the second best results in RMSE, only around 14% worse than the best setup. This results suggests that the method can be specially beneficial when deploying some machine learning algorithm over a dataset from a different area of knowledge where it could be harder to make decisions based on the variable physical meaning.

5.1.4 Impact of Outlier Detection

The deployment of the outlier detection strategies for the target variable, described in 3.3.2, was also evaluated through an experiment where, keeping all things equal, the algorithm was run with the target variable in the dataset being slightly modified by different combinations of all the possibilities of filters (negative values, excessive power and solar hour). The results are compared with a baseline scenario where no filters are applied to the target variable.

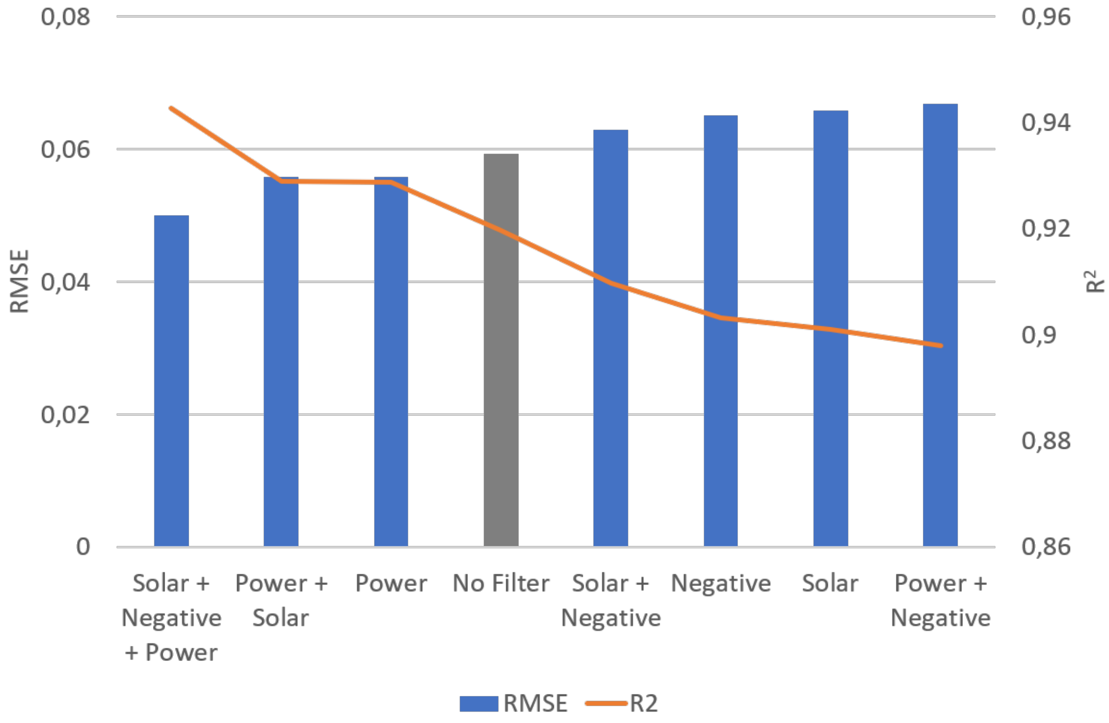


Figure 5.3: Impact of different outlier detection strategies for the target variable.

The first takeaway is that, perhaps contradicting the expectation, the baseline without any filters is not the worse case scenario. The main difference seems to be related with the introduction of the power-based filter which even alone can improve the performance against the baseline. The main conclusion is that using the 3 filter in conjunction provides the best results with a RMSE improvement around 16% when compared to the baseline scenario; additionally, the correlation metric is also the best with this filtering option.

5.1.5 Impact of Post-Processing

As previously discussed in 4.1.4, the conditions applied to filter out values from the target variable, remain valid to be applied on the forecast output. This means that no negative values should be allowed, no production outside solar hours and no power above the rated capacity of the installed capacity. To evaluate the benefits of this approach, two different runs of the model were performed for site 1, using the same network layout and pre-processing techniques; the only difference was running or not the post-processing stage for the outputs of the model.

The results, with an improvement of around 5% in RMSE, clearly state the benefits of conduction a post-processing stage of the results. This underlines the powerful combination of a machine learning approach with human insights that can further constraint and enhance the forecast results.

Metric	**No**	**Yes**	
MAE	0,0321	0,0300	-6,5%
RMSE	0,0564	0,0538	-4,6%
R^2	0,9525	0,9499	-0,3%

Table 5.4: Impact of post-processing techniques in forecast error..

5.2 LSTM Model with Transfer Learning

5.2.1 Site Proximity

The basic concept of transfer learning consists on the idea of taking information from some source domain in order to improve the forecast for the target domain, sharing in this particular case, the same task of solar power production forecast. The results obtained using the methods described in section 4.2.1 are presented in table 5.5.

km	**1**	**2**	**3**	**4**	**5**	**1023**
1		123	38	76	52	342
2			109	60	71	233
3				51	47	339
4					35	293
5						295
1023						

R^2	**1**	**2**	**3**	**4**	**5**	**1023**
1		0,865	0,906	0,891	0,906	0,678
2			0,872	0,898	0,898	0,730
3				0,906	0,917	0,692
4					0,923	0,710
5						0,715
1023						

Table 5.5: Distances and correlation measure for power production between sites.

The main standout from the evaluation of site proximity, either based on distance or correlation criteria, is that site 1023 exhibits smaller correlation with the other parks, particularly when compared with how sites 1-5 relate with each other. The result is not surprising considering the geographical location of site 1023 on the center of the country while the others are more closely located in the south. Judging by the defined criteria, park 1023 should not be used as a source for transfer learning towards the other sites. Regarding the remaining five locations, site 2 , with correlation values below 0.9, also looks slightly displaced when compared to the other four sites, specially site 1 and 3 that are also more than 100km away.

5.2.2 Transfer Learning Approaches

The first experiment was aimed at exploring the different possibilities of transferring the knowledge of pre-trained models to the target. In order to do that, two sites were selected to use as source domains for site 1 forecast: 3 and 5 were selected since they are closer geographically and also exhibit slightly higher correlation coefficient values for their production time-series. All available data from each of the source sites was used; as for the target, it was considered that only one week of data was available at the time of the forecast.

TL Approach	MAE	RMSE	R^2
Feature Extraction	0,0191	0,0386	0,9661
Weight Initialization	0,0197	0,0402	0,9635

Table 5.6: Performance of different transfer learning implementations.

The results show that using feature extraction, where all layers except the final LSTM are frozen, outperforms using source data only for weight initialization. The benefit of using a feature extraction approach seems to be around 4% in RMSE metric.

5.2.3 Performance across Transfer Learning Stages

Looking with further detail into the case of feature extraction, it's also interesting to check how through the transfer learning process, the forecast accuracy evolves. The case here consists of taking one week of data from the target site (1) and evaluating how accurate the different train steps, with different datasets, can forecast solar production based on source sites (3 and 5), identified as closer using either distance or correlation criteria.

The first and most important result, illustrated in figure 5.4, is that the model with transfer learning (denoted 3+5_1) shows the lowest RMSE and the highest correlation coefficient; in fact, when compared with the model only trained with target data, the improvement on RMSE is around 55%. Then, it's also interesting to note that the model trained with only source data can outperform the model trained only with target data by about 41%.

These results are a strong indication of the potential of transfer learning to overcome the data scarcity issues of the cold-start problem.

5.2.4 Dependency on Amount of Target Data

The goal of this work is to explore the potential of transfer learning to improve the forecasts when the target site has very small amounts of data. But, for real world applications, sites start to collect more and more data when they start operation. Then arises the question: for how long

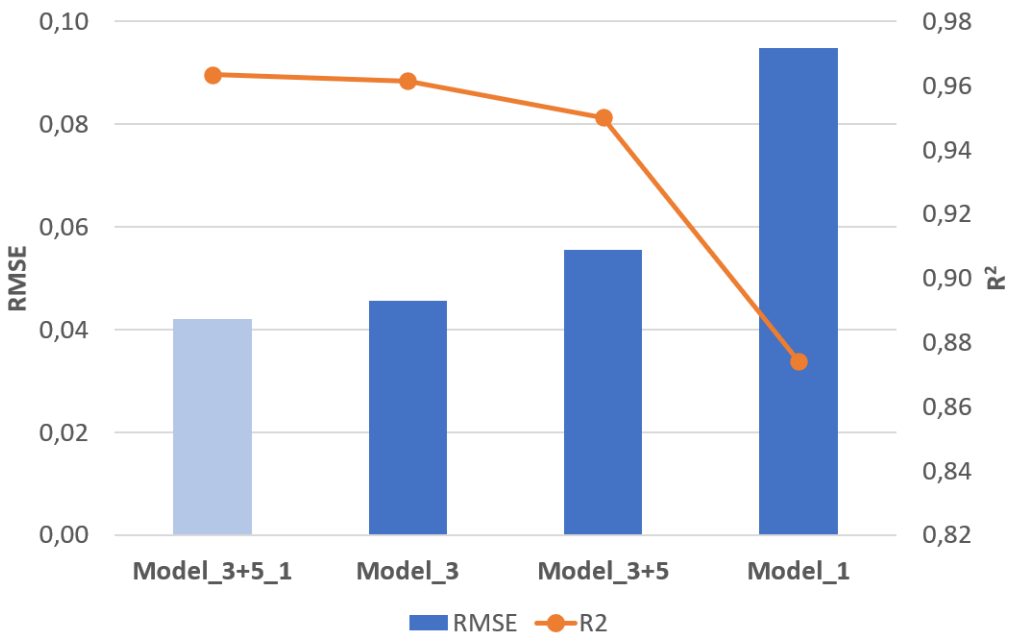


Figure 5.4: Performance of transfer learning model (3+5_1) and another models built through the process.

does it make sense to use a transfer learning model instead of a more standard version where the learning comes all from the target site data? To answer this question, an experiment was devised where, keeping all other settings equal, the only change is the amount of data used to fine-tune the model based on target data.

Theoretically, the expectation would be that for smaller amounts of target data, the model without transfer learning (based only in target data) would be outperformed by the model using the transfer learning paradigm. The plot presented in 5.5 somehow seems to agree with this idea: when the amount of target data is very small (less than 6 months), the model that doesn't use transfer learning shows an error which is roughly two times of what is achieved by the model with transfer learning. This result can be related with the fact that looking at periods so small, there's not enough amount of data to infer all the patterns that may exist throughout the year in solar power production. Looking at the same figure, it looks like there's a cut-off around 6 months of target data: in fact, the RMSE metric looks to be similar with or without transfer learning when the model takes in at least 6 months of data. This suggests such an amount of data is the minimum necessary to capture the required patterns to produce comparable forecasts to a transfer learning model.

It would be interesting to make a similar study for sites located in a region where the weather exhibits less distinct yearly patterns such as the Winter and Summer climate patterns verified in Portugal; the hypothesis is that perhaps for a location at lower latitudes, with smaller weather variation during the year, a smaller amount of data would be necessary for the model without transfer learning to learn all the required patterns to outperform the transfer learning model.

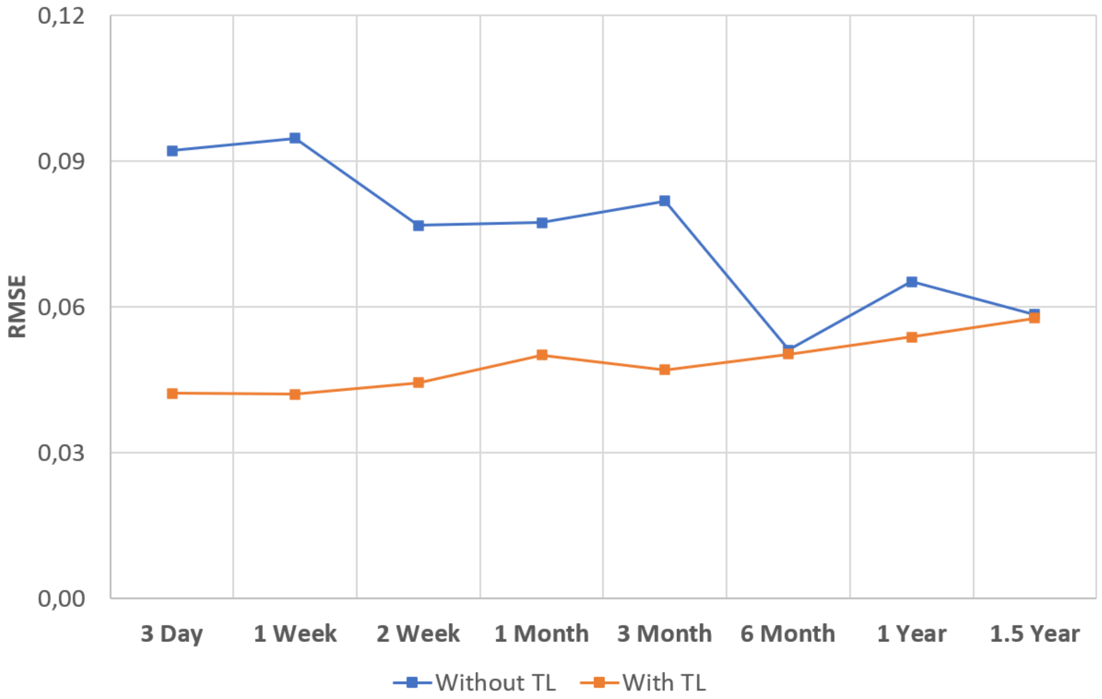


Figure 5.5: Model comparison with different amounts of target data, with and without the application of transfer learning paradigm.

5.3 Estimation of Economical Benefits

The analysis of the forecast error is the typical way to evaluate and guide the optimization process of many machine learning applications. On this real-world example, it's also relevant to highlight how much this error minimization translates into economical benefits. The energy market basic mechanism is described in chapter 1 where the deviations to real production are penalized due to the need of the system operator to intervene in order to balance supply and demand.

A simple exercise was conducted to understand how much benefit can be obtained: considering a 10-days window for site 3 (with one week of available target data and the possibility of transfer learning using sites 4 and 5 as sources), the deviation to real production is calculated in MWh. The assumption is that such deviations will be penalized at spot price market value for each hour, considering daily market prices from Iberian market.

On figure 5.6, it's possible to observe the daily fluctuations of spot prices during the 10-days period. The deviation of the model without transfer learning is higher and, additionally, the larger deviations seems to occur more frequently during the day period where the prices are also typically higher, driven by the demand. This means that deviations during periods of higher prices will have a larger penalty because the prices will be higher. It gives a direct economical incentive to building a model which focus on minimizing RMSE for instance (as opposed to MAE) since that can translate to larger benefits originated from the normal hourly fluctuation of prices.

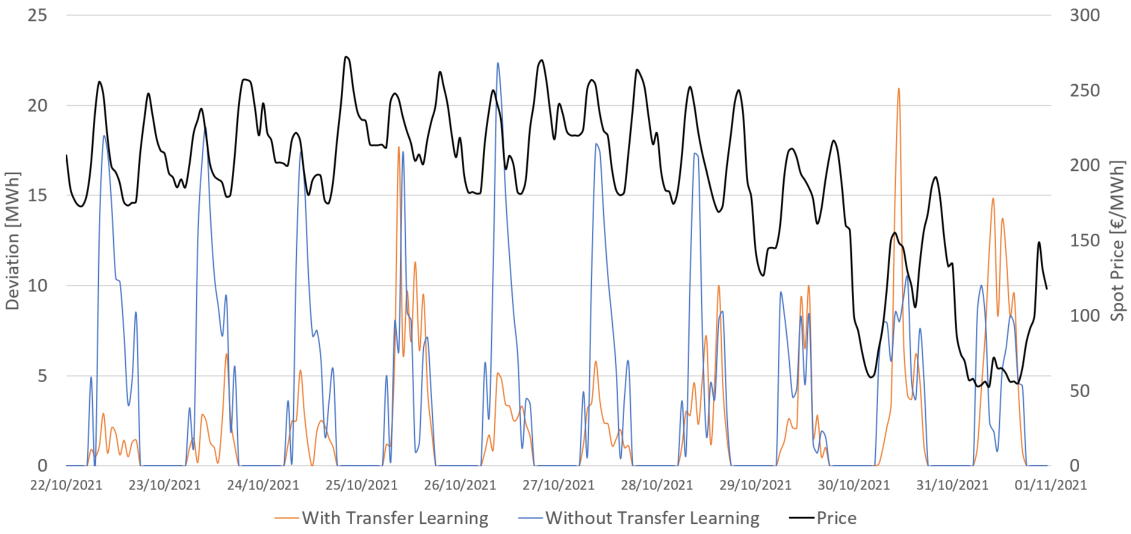


Figure 5.6: Deviation from real production of forecast model with and without transfer learning for site 3; MIBEL spot prices on secondary axis.

Looking with further detail into the results on table 5.7, the model with transfer learning outperforms the standard model only learn from target data by almost 40% in RMSE which is in accordance to results obtained for other sites and presented in previous sections. Then, the sum of all the deviations for the forecast period is almost the double in the case without transfer learning to a value of 890MWh; to put into the site-specific context, the deviation corresponds roughly to 18h of production at the nominal power of the photovoltaic system. For the case with transfer learning, these deviations amount to 461MWh or 9h of production at nominal power. It's also interesting to note that the average spot price, weighted by the production in each period, is approximately 13% lower in the case with transfer learning because this model has smaller deviations during periods of higher prices. This means that the economical benefit will come both from the smaller amount of deviations but also because these deviations will the penalized at a lower average price. Finally, without transfer learning, the deviation to real production would amount to roughly 172m€ which compares to around 77m€ when transfer learning is deployed. The benefit of this approach is a reduction in deviations penalty cost of around 55%.

Model		**Model_3** Without Transfer Learning	**Model_4+5_3** With Transfer Learning	Δ
Error	**MAE**	3,711	1,919	-48%
	RMSE	6,339	3,844	-39%
	R^2	0,827	0,936	13%
Deviation	**MWh**	890	461	-48%
	€/MWh	193 €	168 €	-13%
	€	171 847 €	77 364 €	-55%

Table 5.7: Economical benefit of transfer learning.

Chapter 6

Conclusions

The role of renewable energy for the sustainability of global development and climate change mitigation is the fundamental reason that justifies this work. Solar energy in particular, assumes a special relevance as a feasible solution to satisfy demand proving that the economical challenges of integrating a variable source into the power grid can be solved. The accuracy of forecast models of PV production in solar farms is at the core of its competitiveness since it allows not just to maximize the market value of solar electricity but also to ensure fault detection and optimal scheduling of maintenance activities. The aim of this work was to show that the transfer learning paradigm can be decisive in the specific task of predicting the PV production for sites without representative historical data that could be used to develop traditional data-driven models. The literary review presented in this report was the first step of this master thesis and it reviewed the commonly used models for PV forecast and theoretical background for transfer learning, inspiring the strategy to be followed regarding benchmark models, algorithms, tuning and optimization strategies.

The actual work started with a deeper look at the data structures and its accessibility. The decision was to create a database to simplify the storage of data from different sources, different time horizons and different sites to make sure that data was always available for the desired forecasts. In practice, the database was used to store all the raw data (from Meteo Galicia and production sites) and then also to store a merged version to aggregate both sources of data into unified hourly data. This step was enabled by the use of R for manipulation of data and insertion into database. Reading data into Python, the first step was then to transform the database format into a more typical machine learning dataset with timestamp, predictors and target variable with each row corresponding to one hour of observations.

Before modeling, it was also relevant to go through the exploratory data analysis collecting useful insights to help with the forecast model. The dataset is mostly numerical and the few categorical variables could be discarded due to the unsuitability for time series forecast. Then some other variables were shown to be pointless for instance the ones related with snow precipitation which is thoroughly uncommon in Portugal. Special attention was then paid to the variables related with the radiation flowing in a out of the Earth system. The analysis of the target variable was done considering the known physical characteristics of the production system providing a systematic way of outlier detection in solar power production. Regarding bivariate analysis it showed some severe high correlations for some variables which was expected considering their nature and definition; nevertheless, this was an insight that could be used later in the modeling stage to simplify the dataset and reduce the number of features. The correlation with the target variable is the highest in short-wave radiation flux which makes sense considering that's the kind of radiation targeted by the solar cells.

Due to the potential of real-world application for this work, special attention was paid to the time frame of data availability for some forecast horizon. The conclusion was that it made sense to build the dataset at some specified time only with data that would have been available at that time. Then, it was also important to define some baseline models for a sanity check to verify the superior performance of a LSTM-based model. The LSTM-based model was also able to outperform regression trees, simple RNN and the next-best (previous hour model) by about 20% to 40% when focusing on RMSE. The data pre-processing, before feeding it into the model, was also explored both as a pre-condition for the algorithm to work (scaling and missing values handling) and also to improve the actual results of the forecast (outliers and feature engineering). The design of the forecast model was based on LSTM and dropout layers, linked together on a deep learning network, built within the sequential Keras framework. The best parameters were found through hyperparameter optimization with Keras tuner; it was found that a smaller model with only two LSTM layers was capable of having a better performance when compared with more complex models with extra layers.

The main takeaway from the assessment of feature engineering options was that the feature augmentation technique of adding extra predictors based on timestamp information was key in improving the error metrics. Additionally, it was also shown that these time predictors alone were able to outperform the scenario where only meteorological predictors was used. Finally, using a custom feature selection based on insights taken from exploratory data analysis also showed to be a good decision for error minimization; in fact, the approach combining physical knowledge of the photovoltaic and meteorological systems was able to outperform automated feature selection techniques such as the XGBoost by roughly 14% in RMSE. Nevertheless, these automated approaches remain essential to deal with problems where deep knowledge of the problem variables is not available.

The impact of using different outlier detection strategies, for the target variable, is also quantified and the conclusion is that the best setup consists in the joint use of filters for solar hour, negative values and maximum installed power. The RMSE improvement,when compared to the baseline scenario without any filtering on the target variable, was estimated to be around 16%. Additionally, applying the same techniques as post-processing stage to the forecast, was able to improve the RMSE error by about 5%, further underlining the benefits of combining the power of artificial intelligence and human insights into the data.

The application of transfer learning was tested in different scenarios. First, two different approaches were tried with the network being used both for feature extraction and weight initialization. The results obtained suggest that the feature extraction approach provides a benefit of around 4% in the RMSE error metric for the forecast. Additionally, it was also found that the weight initialization approach may be overly sensitive to target data. So, feature extraction was favored and used going forward.

During the transfer learning process, different models are developed corresponding to each stage: first trained with source sites and then fine-tuned with target. It is then interesting to compare the performance across these stages. It was found that the transfer learning model can outperform the model trained only on target data by 55% when considering a cold-start scenario where only 1 week of data is available for the target. Additionally the model trained only on source sites can also outperform target-only model by roughly 41% in the same scenario which suggests that on a pure cold-start (with no data from target), a good backup solution can be found in a model trained with selected sources.

A major question about the applicability of transfer learning to the cold-start problem is the time frame where these models can provide a relevant improvement to traditional models. The results from this work suggest that for the first six months of operation, the benefit of transfer learning will be quite high; after that period, the collected data should allow a traditional machine learning model, trained only on target data, to produce similar results.

Finally, the economical benefit of the transfer learning approach was also quantified with a simple approach where the deviations of the forecast towards the real production was penalized with Iberian marker spot prices. The results show that the benefits of transfer learning are originated from two effects. First, since the deviation during peak hours (with typically higher prices) is smaller, the average price for each MWh of deviation is roughly 13% smaller in transfer learning scenario. Second, the total amount of deviation is roughly 48% smaller with transfer learning. Together, these two effects translate into a reduction of the penalties around 55% when transfer learning is deployed.

Future works could focus on further exploration of the machine learning model trying for instance different tuners to search different configurations and also the exploration of different loss functions for instance; it would be interesting to find how much the model should be tailored for specific sites or whether the configuration is more site independent. On the other hand, this work suggests the first six months of operation to be the crucial period where transfer learning models can bring more benefit; nevertheless, maybe this period can differ depending on site location, for instance, so that could also be a direction for another study.

Bibliography

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., … Isard, M. (2016). Tensorflow: A system for large-scale machine learning [Journal Article]. *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 265-283.

Abdulrahman, M. L., Ibrahim, K. M., Gital, A. Y., Zambuk, F. U., Ja'afaru, B., Yakubu, Z. I., & Ibrahim, A. (2021). A review on deep learning with focus on deep recurrent neural network for electricity forecasting in residential building [Journal Article]. *Procedia Computer Science*, *193*, 141-154. Retrieved from https://www.sciencedirect.com/science/article/pii/S187705092102055X doi: https://doi.org/10.1016/j.procs.2021.10.014

Aguilar, C., Munoz-Romero, S., & Rojo-Álvarez, J. L. (2020). Cold-start promotional sales forecasting through gradient boosted-based contrastive explanations [Journal Article]. *IEEE Access*, *PP*, 1-1. doi: 10.1109/ACCESS.2020.3012032

Al-Yahyai, S., Charabi, Y., & Gastli, A. (2010). Review of the use of numerical weather prediction (nwp) models for wind energy assessment [Journal Article]. *Renewable and Sustainable Energy Reviews*, *14*(9), 3192-3198. Retrieved from https://www.sciencedirect.com/science/article/pii/S1364032110001814 doi: https://doi.org/10.1016/j.rser.2010.07.001

Anees, A. S. (2012). Grid integration of renewable energy sources: Challenges, issues and possible solutions [Journal Article]. *2012 IEEE 5th India International Conference on Power Electronics (IICPE)*, 1-6. doi: 10.1109/IICPE.2012.6450514

Bourdeau, M., Zhai, X. q., Nefzaoui, E., Guo, X., & Chatellier, P. (2019). Modeling and forecasting building energy consumption: A review of data-driven techniques [Journal Article]. *Sustainable Cities and Society*, *48*, 101533. Retrieved from https://www.sciencedirect.com/science/article/pii/S2210670718323862 doi: https://doi.org/10.1016/j.scs.2019.101533

Chapelle, O., Schölkopf, B., & Zien, A. (2006). *Semi-supervised learning* [Book]. The MIT Press.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system [Conference Proceedings]. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (p. 785-794).

Chollet, F. (2021). *Deep learning with python* [Book]. Simon and Schuster.

Daumé III, H. (2009). Frustratingly easy domain adaptation [Journal Article]. *arXiv preprint arXiv:0907.1815*.

Duan, L., Xu, D., & Tsang, I. W. (2012). Domain adaptation from multiple sources: A domain-dependent regularization approach [Journal Article]. *IEEE Transactions on Neural Networks and Learning Systems*, *23*(3), 504-518. doi: 10.1109/TNNLS.2011.2178556

Fan, C., Sun, Y., Xiao, F., Ma, J., Lee, D., Wang, J., & Tseng, Y. C. (2020). Statistical investigations of transfer learning-based methodology for short-term building energy predictions [Journal Article]. *Applied Energy*, *262*. Retrieved from https://search.ebscohost.com/login.aspx?direct=true&db=edselp&AN=S0306261920300118&site=eds-live doi: 10.1016/j.apenergy.2020.114499

Fang, X., Gong, G., Li, G., Chun, L., Li, W., & Peng, P. (2021). A hybrid deep transfer learning strategy for short term cross-building energy prediction [Journal Article]. *Energy*, *215*, 119208. Retrieved from https://www.sciencedirect.com/science/article/pii/S036054422032315X doi: https://doi.org/10.1016/j.energy.2020.119208

Ferreira, M., Santos, A., & Lucio, P. (2019). Short-term forecast of wind speed through mathematical models [Journal Article]. *Energy Reports*, *5*, 1172-1172-1184. Retrieved from https://search.ebscohost.com/login.aspx?direct=true&db=edselp&AN=S2352484718303275&site=eds-live doi: 10.1016/j.egyr.2019.05.007

Fetting, C. (2020). The european green deal [Journal Article].

Florian, Z. (2020). Load nowcasting: Predicting actuals with limited data [Journal Article]. *Energies*, *13*(6), 1443-1443-1443. Retrieved from https://doaj.org/article/83f764e099cb4e28ae3a7a69e4835b4e doi: 10.3390/en13061443

Foucquier, A., Robert, S., Suard, F., Stéphan, L., & Jay, A. (2013). State of the art in building modelling and energy performances prediction: A review [Journal Article]. *Renewable and Sustainable Energy Reviews*, *23*, 272-288. Retrieved from https://www.sciencedirect.com/science/article/pii/S1364032113001536 doi: https://doi.org/10.1016/j.rser.2013.03.004

Gao, Y., Ruan, Y., Fang, C., & Yin, S. (2020). Deep learning and transfer learning models of energy consumption forecasting for a building with poor information data [Journal Article]. *Energy and Buildings*, *223*, N.PAG. Retrieved from https://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=145407672&site=eds-live doi: 10.1016/j.enbuild.2020.110156

Ghifary, M., Kleijn, W., & Zhang, M. (2014). *Domain adaptive neural networks for object recognition* [Book]. doi: 10.1007/978-3-319-13560-1_76

Giebel, G., Brownsword, R., Kariniotakis, G., Denhard, M., & Draxl, C. (2011). *The state of the art in short-term prediction of wind power a literature overview, 2nd edition* [Book]. doi: 10.13140/RG.2.1.2581.4485

Gurupira, T., & Rix, A. (2016). Photovoltaic system modelling using pvlib-python [Journal Article].

Hassan, Q., Jaszczur, M., & Przenzak, E. (2017). Mathematical model for the power generation from arbitrarily oriented photovoltaic panel [Journal Article]. *E3S web of conferences*, *14*, 01028.

Hirth, L. (2013). The market value of variable renewables: The effect of solar wind power variability on their relative price [Journal Article]. *Energy Economics*, *38*, 218-236. Retrieved from https://www.sciencedirect.com/science/article/pii/S0140988313000285 doi: https://doi.org/10.1016/j.eneco.2013.02.004

Hooshmand, A., & Sharma, R. (2019). Energy predictive models with limited data using transfer learning [Journal Article]. *e-Energy '19*, 12-16. doi: 10.1145/3307772.3328284

Hu, Q., Zhang, R., & Zhou, Y. (2016). Transfer learning for short-term wind speed prediction with deep neural networks [Journal Article]. *Renewable Energy*, *85*, 83-95. Retrieved from https://www.sciencedirect.com/science/article/pii/S0960148115300574 doi: https://doi.org/10.1016/j.renene.2015.06.034

Huiting, Z., Yuan, J., & Chen, L. (2017). Short-term load forecasting using emd-lstm neural networks with a xgboost algorithm for feature importance evaluation [Journal Article]. *Energies*, *10*, 1168. doi: 10.3390/en10081168

Jiang, J., & Zhai, C. (2007). Instance weighting for domain adaptation in nlp [Journal Article]. *Proceedings of the 45th annual meeting of the association of computational linguistics*, 264-271.

Jihoon, M., Junhong, K., Pilsung, K., & Eenjun, H. (2020). Solving the cold-start problem in short-term load forecasting using tree-based methods [Journal Article]. *Energies*, *13*(4), 886-886-886. Retrieved from https://doaj.org/article/ba11e32597de42148c34b9a5cfedd182 doi: 10.3390/en13040886

Kumaraswamy, R., Odom, P., Kersting, K., Leake, D., & Natarajan, S. (2015). *Transfer learning via relational type matching* [Book]. doi: 10.1109/ICDM.2015.138

LeDell, E., & Poirier, S. (2020). H2o automl: Scalable automatic machine learning [Conference Proceedings]. In *Proceedings of the automl workshop at icml* (Vol. 2020).

Leva, S., Mussetta, M., & Ogliari, E. (2018). Pv module fault diagnosis based on microconverters and day-ahead forecast [Journal Article]. *IEEE Transactions on Industrial Electronics*, *PP*, 1-1. doi: 10.1109/TIE.2018.2879284

Li, A., Xiao, F., Fan, C., & Hu, M. (2021). Development of an ann-based building energy model for information-poor buildings using transfer learning [Journal Article]. *Building Simulation*, *14*(1), 89-89-101. Retrieved from https://search.ebscohost.com/login.aspx?direct=true&db=edb&AN=147198985&site=eds-live doi: 10.1007/s12273-020-0711-5

Li, J., Huang, R., He, G., Wang, S., Li, G., & Li, W. (2020). A deep adversarial transfer learning network for machinery emerging fault detection [Journal Article]. *IEEE Sensors Journal*, *20*(15), 8413-8422. doi: 10.1109/JSEN.2020.2975286

Lika, B., Kolomvatsos, K., & Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems [Journal Article]. *Expert Systems with Applications*, *41*(4, Part 2), 2065-2073. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417413007240 doi: https://doi.org/10.1016/j.eswa.2013.09.005

Ma, T., Yang, H., & Lu, L. (2014). Solar photovoltaic system modeling and performance prediction [Journal Article]. *Renewable and Sustainable Energy Reviews*, *36*, 304-315. Retrieved from https://www.sciencedirect.com/science/article/pii/S1364032114002950 doi: https://doi.org/10.1016/j.rser.2014.04.057

Monteiro, C., Fernandez-Jimenez, L. A., Ramirez-Rosado, I. J., Muñoz-Jimenez, A., & Lara-Santillan, P. M. (2013). Short-term forecasting models for photovoltaic plants: Analytical versus soft-computing techniques [Journal Article]. *Mathematical Problems in Engineering*, 1-1-9. Retrieved from https://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=94813893&site=eds-live doi: 10.1155/2013/767284

O'Malley, T., Bursztein, E., Long, J., Chollet, F., Jin, H., Invernizzi, L., et al. (2019). *Kerastuner*. https://github.com/keras-team/keras-tuner.

Owusu, P. A., & Asumadu-Sarkodie, S. (2016). A review of renewable energy sources, sustainability issues and climate change mitigation [Journal Article]. *Cogent Engineering*, *3*(1), 1167990. Retrieved from https://doi.org/10.1080/23311916.2016.1167990 doi: 10.1080/23311916.2016.1167990

Pan, S. J., & Yang, Q. (2009). A survey on transfer learning [Journal Article]. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345-1359.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Dubourg, V. (2011). Scikit-learn: Machine learning in python [Journal Article]. *the Journal of machine Learning research*, *12*, 2825-2830.

Perpiñán, O., & Almeida, M. P. (2021). meteoForecast [Computer software manual]. Retrieved from https://github.com/oscarperpinan/meteoForecast/ (R package version 0.54)

R Core Team. (2020). R: A language and environment for statistical computing [Computer software manual]. Vienna, Austria. Retrieved from https://www.R-project.org/

Ribeiro, M., Grolinger, K., ElYamany, H. F., Higashino, W. A., & Capretz, M. A. M. (2018). Transfer learning with seasonal and trend adjustment for cross-building energy forecasting [Journal Article]. *Energy and Buildings*, *165*, 352-363. Retrieved from https://www.sciencedirect.com/science/article/pii/S0378778817329171 doi: https://doi.org/10.1016/j.enbuild.2018.01.034

Rosenstein, M., Marx, Z., Kaelbling, L., & Dietterich, T. (2005). To transfer or not to transfer [Journal Article].

Sala, S., Amendola, A., Leva, S., Mussetta, M., Niccolai, A., & Ogliari, E. (2019). Comparison of data-driven techniques for nowcasting applied to an industrial-scale photovoltaic plant [Journal Article]. *Energies*, *12*. doi: 10.3390/en12234520

Seung-Min, J., Sungwoo, P., Seung-Won, J., & Eenjun, H. (2020). Monthly electric load forecasting using transfer learning for smart cities [Journal Article]. *Sustainability*, *12*(6364), 6364-6364-6364. Retrieved from https://doaj.org/article/5c60c8fc207140a1adfabe079b62ea59 doi: 10.3390/su12166364

Stanev, R., & Tanev, T. (2018). Mathematical model of photovoltaic power plant [Conference Proceedings]. Institute of Electrical and Electronics Engineers Inc. Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-85053837263&doi=10.1109%2fSIELA.2018.8447173&partnerID=40&md5=f2ae5eac135cb04df462754e79a6233e doi: 10.1109/SIELA.2018.8447173

Wang, Z., & Srinivasan, R. S. (2017). A review of artificial intelligence based building energy use prediction: Contrasting the capabilities of single and ensemble prediction models [Journal Article]. *Renewable and Sustainable Energy Reviews*, *75*, 796-808. Retrieved from https://www.sciencedirect.com/science/article/pii/S1364032116307420 doi: https://doi.org/10.1016/j.rser.2016.10.079

Wei, Y., Zhang, X., Shi, Y., Xia, L., Pan, S., Wu, J., … Zhao, X. (2018). A review of data-driven approaches for prediction and classification of building energy consumption [Journal Article]. *Renewable and Sustainable Energy Reviews*, *82*, 1027-1047. Retrieved from https://www.sciencedirect.com/science/article/pii/S136403211731362X doi: https://doi.org/10.1016/j.rser.2017.09.108

Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). A survey of transfer learning [Journal Article]. *Journal of Big Data*, *3*(1), 9. Retrieved from https://doi.org/10.1186/s40537-016-0043-6 doi: 10.1186/s40537-016-0043-6

Wolf, S., Teitge, J., Mielke, J., Schütze, F., & Jaeger, C. (2021). The european green deal — more than climate neutrality [Journal Article]. *Intereconomics*, *56*(2), 99-107. Retrieved from https://doi.org/10.1007/s10272-021-0963-z doi: 10.1007/s10272-021-0963-z

Xiao, T., Xu, P., He, R., & Sha, H. (2022). Status quo and opportunities for building energy prediction in limited data context—overview from a competition [Journal Article]. *Applied Energy*, *305*. Retrieved from https://search.ebscohost.com/login.aspx?direct=true&db=edselp&AN=S0306261921011570&site=eds-live doi: 10.1016/j.apenergy.2021.117829

Xie, C., Tank, A., Greaves-Tunnell, A., & Fox, E. (2017). A unified framework for long range and cold start forecasting of seasonal profiles in time series [Journal Article].

Xu, C., Tao, D., & Xu, C. (2013). A survey on multi-view learning [Journal Article]. *arXiv preprint arXiv:1304.5634*.

Zhang, Y., & Luo, G. (2015). Short term power load prediction with knowledge transfer [Journal Article]. *Information Systems*, *53*, 161-169. Retrieved from https://www.sciencedirect.com/science/article/pii/S0306437915000150 doi: https://doi.org/10.1016/j.is.2015.01.005

Zhang, Y., & Yang, Q. (2018). An overview of multi-task learning [Journal Article]. *National Science Review*, *5*(1), 30-43.

Zhou, S., Zhou, L., Mao, M., & Xi, X. (2020). Transfer learning for photovoltaic power forecasting with long short-term memory neural network [Journal Article]. , 125-125-132. Retrieved from https://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.9070676&site=eds-live doi: 10.1109/BigComp48618.2020.00-87

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., … He, Q. (2020). A comprehensive survey on transfer learning [Journal Article]. *Proceedings of the IEEE*, *PP*, 1-34. doi: 10.1109/JPROC.2020.3004555

Appendix

A Description of Dataset Variables

Variable	Source	Description	Type	Unit
timestamp	-	Timestamp of observation in GMT +0h.	timestamp	-
cape	Meteo Galicia	Convective available potential energy.	Numeric	[J/kg]
cfh	Meteo Galicia	Cloud cover at high levels.	Numeric	[0-1]
cfl	Meteo Galicia	Cloud cover at low levels.	Numeric	[0-1]
cfm	Meteo Galicia	Cloud cover at mid levels.	Numeric	[0-1]
cft	Meteo Galicia	Cloud cover at low and mid levels.	Numeric	[0-1]
cin	Meteo Galicia	Convective inhibition.	Numeric	-
conv_prec	Meteo Galicia	Total accumulated convective rainfall between each model output.	Numeric	[kg/m^2]
dir	Meteo Galicia	Wind direction at 10m.	Numeric	[º]
HGT500	Meteo Galicia	Geopotential height at 500mb.	Numeric	[m]
HGT850	Meteo Galicia	Geopotential height at 850mb.	Numeric	[m]
HGTlev1	Meteo Galicia	Geopotential height at model level 1.	Numeric	[m]
HGTlev2	Meteo Galicia	Geopotential height at model level 2.	Numeric	[m]
HGTlev3	Meteo Galicia	Geopotential height at model level 3.	Numeric	[m]
land_use	Meteo Galicia	Land Use/Vegetation Type (constant value for each site).	Categorical	-
lhflx	Meteo Galicia	Surface downward latent heat flux.	Numeric	[W/m^2]
lwflx	Meteo Galicia	Surface downwelling longwave flux.	Numeric	[W/m^2]
lwm	Meteo Galicia	Land/water mask (constant value for all sites, 0=land).	Categorical	-
meteograms	Meteo Galicia	Meteograms (plot of meteorological variables over time).	Categorical	-
mod	Meteo Galicia	Wind module at 10m.	Numeric	[m/s]
mslp	Meteo Galicia	Mean sea level pressure.	Numeric	[Pa]
pbl_height	Meteo Galicia	PBL Height.	Numeric	[m]
prec	Meteo Galicia	Total accumulated rainfall between each model output.	Numeric	[kg/m^2]
rh	Meteo Galicia	Relative humidity at 2m.	Numeric	[0-1]
shflx	Meteo Galicia	Surface downward sensible heat flux.	Numeric	[W/m^2]
snow_prec	Meteo Galicia	Total accumulated large scale snowfall between each model output.	Numeric	[kg/m^2]
snowlevel	Meteo Galicia	Snow level.	Numeric	[m]
sst	Meteo Galicia	Sea surface temperature.	Numeric	[K]
swflx	Meteo Galicia	Surface downwelling shortwave flux.	Numeric	[W/m^2]
T500	Meteo Galicia	Temperature at 500mb.	Numeric	[K]
T850	Meteo Galicia	Temperature at 850mb.	Numeric	[K]
temp	Meteo Galicia	Temperature at 2m.	Numeric	[K]
topo	Meteo Galicia	Topography (Constant value for each site - altitude).	Numeric	[m]
u	Meteo Galicia	Lon-wind at 10m.	Numeric	[m/s]
ulev1	Meteo Galicia	Lon-wind at model level 1.	Numeric	[m/s]
ulev2	Meteo Galicia	Lon-wind at model level 2.	Numeric	[m/s]
ulev3	Meteo Galicia	Lon-wind at model level 3.	Numeric	[m/s]
v	Meteo Galicia	Lat-wind at 10m.	Numeric	[m/s]
visibility	Meteo Galicia	Visibility.	Numeric	[m]
vlev1	Meteo Galicia	Lat-wind at model level 1.	Numeric	[m/s]
vlev2	Meteo Galicia	Lat-wind at model level 2.	Numeric	[m/s]
vlev3	Meteo Galicia	Lat-wind at model level 3.	Numeric	[m/s]
weasd	Meteo Galicia	Water Equivalent of Accumulated Snow Depth (zero in all sites).	Numeric	[kg/m^2]
wind_gust	Meteo Galicia	Wind gust.	Numeric	[m/s]
y_production_mw	Smartwatt	Production of power plant.	Numeric	[MW]

Table 6.1: Detailed description of the variables used for solar power production forecast.

B Univariate Analysis

Variable	Counters				Location								Dispersion			Shape	
	Count	Count Distinct	Missing Values [#]	Missing Values [%]	Mean	Min	Q1	Median	Q3	Max	Range	IQR	Standard Deviation	Variance	Variation Coefficient	Skewness	Kurtosis
cape	16053	3479	0	0,00%	56,25	0,00	0,00	0,00	3,80	2383	2383	3,80	188,90	35685	336%	5,03	30,73
cfh	16053	208	0	0,00%	0,09	0,00	0,00	0,00	0,10	1,00	1,00	0,10	0,18	0,03	201%	2,65	7,85
cfl	16053	2444	0	0,00%	0,09	0,00	0,00	0,00	0,01	1,00	1,00	0,01	0,18	0,03	210%	2,14	3,65
cfm	16053	174	0	0,00%	0,03	0,00	0,00	0,00	0,00	1,00	1,00	0,00	0,10	0,01	357%	4,86	27,76
cft	16053	2060	0	0,00%	0,16	0,00	0,00	0,00	0,30	1,00	1,00	0,30	0,23	0,05	143%	1,49	1,66
cin	16053	4449	0	0,00%	-21,26	-1288,24	-0,03	0,00	0,00	0,05	1288	0,03	86,02	7399	-405%	-5,93	44,11
conv_prec	16053	1893	0	0,00%	0,02	0,00	0,00	0,00	0,00	5,92	5,92	0,00	0,20	0,04	912%	14,30	266,28
dir	16053	16030	0	0,00%	232,63	0,01	150,75	279,95	310,74	359,99	359,98	159,99	99,26	9852	43%	-0,81	-0,58
HGT500	16053	15717	0	0,00%	5693	5246	5622	5712	5781	5930	683,61	158,69	114,74	13166	2,02%	-0,71	0,06
HGT850	16053	15346	0	0,00%	1504	1318	1476	1510	1538	1626	307,31	61,65	49,16	2417	3,27%	-0,57	0,33
HGTlev1	16053	14960	0	0,00%	187,35	185,58	186,93	187,32	187,75	189,14	3,56	0,82	0,61	0,37	0,32%	0,19	-0,28
HGTlev2	16053	15147	0	0,00%	229,96	225,28	228,85	229,87	231,02	234,65	9,37	2,17	1,60	2,56	0,70%	0,20	-0,27
HGTlev3	16053	15493	0	0,00%	264,21	257,31	262,55	264,07	265,78	271,22	13,92	3,22	2,37	5,61	0,90%	0,21	-0,26
land_use	16053	1	0	0,00%	2,00	2,00	2,00	2,00	2,00	2,00	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
lhflx	16053	14375	0	0,00%	67,51	-16,06	4,60	22,28	106,10	405,20	421,26	101,50	88,65	7860	131%	1,41	0,93
lwflx	16053	16019	0	0,00%	327,05	215,01	299,71	327,53	355,11	447,02	232,01	55,40	37,80	1429	12%	-0,08	-0,49
lwm	16053	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
meteograms	16053	13	0	0,00%	102,05	101,00	101,00	101,00	103,00	119,00	18,00	2,00	2,82	7,98	2,77%	4,57	22,40
mod	16053	16048	0	0,00%	3,24	0,01	1,86	3,01	4,34	12,27	12,26	2,48	1,84	3,38	57%	0,76	0,61
mslp	16053	15359	0	0,00%	101833	99909	101478	101794	102178	103463	3554	700,23	556,73	309950	0,55%	0,12	0,22
pbl_height	16053	16050	0	0,00%	421,74	23,97	106,64	258,59	600,86	2400	2376	494,21	419,47	175952	99%	1,39	1,38
prec	16053	1810	0	0,00%	0,04	-0,04	0,00	0,00	0,00	13,10	13,14	0,00	0,35	0,12	844%	18,00	448,01
rh	16053	15422	0	0,00%	0,80	0,29	0,67	0,85	0,95	1,00	0,71	0,27	0,18	0,03	22%	-0,79	-0,49
shflx	16053	16005	0	0,00%	40,74	-139,73	-15,74	-4,89	69,03	404,68	544,41	84,77	93,77	8793	230%	1,59	1,64
snow_prec	16053	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
snowlevel	16053	16048	0	0,00%	2481	419,46	1941	2429	3027	4327	3908	1086	714,79	510930	29%	0,05	-0,67
sst	16053	15740	0	0,00%	289,84	270,40	284,51	289,03	294,45	311,62	41,22	9,94	7,35	54,05	2,54%	0,39	-0,28
swflx	16053	6437	0	0,00%	209,69	0,00	0,00	0,00	382,80	1031	1031	382,80	308,85	95390	147%	1,26	0,16
T500	16053	15762	0	0,00%	259,20	241,27	255,15	259,67	263,33	271,28	30,01	8,18	5,30	28,05	2,04%	-0,31	-0,46
T850	16053	15810	0	0,00%	284,40	267,92	280,07	283,86	288,65	301,76	33,84	8,58	5,70	32,50	2,00%	0,18	-0,48
temp	16053	15727	0	0,00%	289,43	270,82	284,64	288,86	293,62	309,47	38,65	8,98	6,71	45,06	2,32%	0,32	-0,25
topo	16053	1	0	0,00%	161,84	161,84	161,84	161,84	161,84	161,84	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
u	16053	16012	0	0,00%	1,18	-8,51	-0,50	1,33	2,88	11,96	20,47	3,38	2,45	6,02	209%	-0,11	0,15
ulev1	16053	16020	0	0,00%	1,49	-9,91	-0,68	1,78	3,64	14,08	24,00	4,33	3,00	9,02	202%	-0,17	-0,10
ulev2	16053	16027	0	0,00%	1,63	-10,99	-0,99	1,87	4,29	15,80	26,79	5,28	3,61	13,05	222%	-0,13	-0,28
ulev3	16053	16020	0	0,00%	1,61	-11,45	-1,25	1,81	4,51	16,56	28,01	5,77	3,96	15,72	246%	-0,11	-0,31
v	16053	16012	0	0,00%	-0,60	-11,22	-2,17	-0,91	0,77	10,82	22,03	2,94	2,47	6,10	-410%	0,55	0,94
visibility	16053	10153	0	0,00%	22252	15,92	24038	24042	24057	24235	24219	19,01	6274	39362879	28%	-3,19	8,23
vlev1	16053	16015	0	0,00%	-0,75	-13,33	-2,76	-1,19	1,05	12,85	26,18	3,81	3,02	9,15	-402%	0,56	0,64
vlev2	16053	16023	0	0,00%	-1,03	-15,72	-3,66	-1,68	1,31	15,81	31,53	4,97	3,71	13,76	-358%	0,60	0,33
vlev3	16053	16036	0	0,00%	-1,24	-17,06	-4,28	-1,99	1,47	16,85	33,90	5,75	4,15	17,19	-334%	0,61	0,15
weasd	16053	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
wind_gust	16053	16051	0	0,00%	4,87	0,02	2,63	4,22	6,55	21,79	21,77	3,92	3,05	9,32	63%	1,05	1,20
y	16053	3469	0	0,00%	0,15	0,00	0,00	0,00	0,28	0,62	0,62	0,28	0,20	0,04	141%	1,11	-0,31

Table 6.2: Univariate analysis for site 1.

Variable	Counters				Location								Dispersion			Shape	
	Count	Count Distinct	Missing Values [#]	Missing Values [%]	Mean	Min	Q1	Median	Q3	Max	Range	IQR	Standard Deviation	Variance	Variation Coefficient	Skewness	Kurtosis
cape	16078	3543	0	0,00%	47,11	0,00	0,00	0,00	6,76	2182	2182	6,76	159,43	25417	338%	5,43	36,66
cfh	16078	210	0	0,00%	0,09	0,00	0,00	0,00	0,10	1,00	1,00	0,10	0,17	0,03	198%	2,71	8,52
cfl	16078	2995	0	0,00%	0,11	0,00	0,00	0,00	0,15	1,00	1,00	0,15	0,20	0,04	178%	1,64	1,45
cfm	16078	184	0	0,00%	0,03	0,00	0,00	0,00	0,00	1,00	1,00	0,00	0,12	0,01	336%	4,40	21,84
cft	16078	2592	0	0,00%	0,18	0,00	0,00	0,05	0,32	1,00	1,00	0,32	0,23	0,05	130%	1,24	0,80
cin	16078	4665	0	0,00%	-18,02	-1339,90	-0,04	0,00	0,00	0,05	1340	0,04	81,74	6681	-454%	-7,20	66,37
conv_prec	16078	1970	0	0,00%	0,03	0,00	0,00	0,00	0,00	5,66	5,66	0,00	0,24	0,06	716%	9,98	125,48
dir	16078	16058	0	0,00%	207,21	0,02	108,35	226,99	322,74	359,99	359,96	214,38	116,40	13550	56%	-0,31	-1,32
HGT500	16078	15745	0	0,00%	5686	5255	5614	5706	5772	5918	662,72	158,08	115,32	13298	2,03%	-0,69	0,00
HGT850	16078	15440	0	0,00%	1503	1327	1473	1509	1538	1637	309,83	65,27	51,36	2638	3,42%	-0,56	0,20
HGTlev1	16078	14811	0	0,00%	175,20	173,61	174,82	175,16	175,54	176,84	3,23	0,72	0,54	0,29	0,31%	0,29	-0,13
HGTlev2	16078	15157	0	0,00%	217,73	213,49	216,74	217,62	218,64	222,01	8,52	1,90	1,43	2,04	0,66%	0,30	-0,12
HGTlev3	16078	15422	0	0,00%	251,93	245,52	250,46	251,74	253,27	258,29	12,77	2,81	2,11	4,46	0,84%	0,30	-0,11
land_use	16078	1	0	0,00%	11,00	11,00	11,00	11,00	11,00	11,00	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
lhflx	16078	14434	0	0,00%	67,57	-10,02	4,88	24,97	104,30	408,60	418,62	99,42	86,72	7520	128%	1,42	1,02
lwflx	16078	16045	0	0,00%	327,13	219,65	297,51	326,32	358,70	424,82	205,17	61,19	39,47	1558	12%	-0,04	-0,72
lwm	16078	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
meteograms	16078	13	0	0,00%	102,67	101,00	101,00	101,00	103,00	119,00	18,00	2,00	3,87	15,01	3,77%	3,20	9,63
mod	16078	16069	0	0,00%	2,98	0,02	1,90	2,86	3,88	12,65	12,64	1,98	1,51	2,29	51%	0,65	0,67
mslp	16078	15374	0	0,00%	101845	99773	101494	101819	102199	103506	3733	705,14	568,22	322872	0,56%	0,03	0,22
pbl_height	16078	16074	0	0,00%	394,18	24,16	99,74	251,96	570,75	2406	2382	471,01	385,52	148623	98%	1,37	1,47
prec	16078	2022	0	0,00%	0,07	-0,05	0,00	0,00	0,00	18,00	18,05	0,00	0,46	0,21	626%	14,92	335,51
rh	16078	14942	0	0,00%	0,82	0,32	0,70	0,87	0,96	1,00	0,68	0,26	0,17	0,03	20%	-0,81	-0,47
shflx	16078	16053	0	0,00%	36,32	-222,34	-27,80	-6,83	66,21	431,91	654,25	94,01	105,39	11107	290%	1,55	1,67
snow_prec	16078	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
snowlevel	16078	16069	0	0,00%	2418	374,34	1894	2359	2942	4213	3838	1048	701,20	491686	29%	0,08	-0,67
sst	16078	15731	0	0,00%	288,89	272,21	284,50	287,98	292,60	308,44	36,23	8,11	6,40	40,97	2,22%	0,48	-0,07
swflx	16078	6402	0	0,00%	202,01	0,00	0,00	0,00	355,12	1023	1023	355,12	302,12	91279	150%	1,31	0,30
T500	16078	15843	0	0,00%	258,96	240,73	254,82	259,36	263,09	270,98	30,25	8,27	5,31	28,20	2,05%	-0,28	-0,47
T850	16078	15825	0	0,00%	283,85	267,21	279,68	283,16	287,94	300,61	33,39	8,26	5,63	31,66	1,98%	0,21	-0,55
temp	16078	15711	0	0,00%	288,81	272,66	284,61	288,04	292,40	307,46	34,80	7,79	6,09	37,08	2,11%	0,45	-0,05
topo	16078	1	0	0,00%	149,72	149,72	149,72	149,72	149,72	149,72	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
u	16078	16033	0	0,00%	0,31	-7,49	-1,06	0,38	1,69	10,16	17,64	2,75	2,11	4,47	674%	-0,03	0,14
ulev1	16078	16045	0	0,00%	0,35	-9,88	-1,55	0,56	2,27	12,94	22,82	3,82	2,86	8,18	810%	-0,09	-0,05
ulev2	16078	16045	0	0,00%	0,60	-11,96	-1,51	0,91	3,00	15,41	27,36	4,51	3,54	12,56	594%	-0,29	0,05
ulev3	16078	16043	0	0,00%	0,71	-13,12	-1,48	1,08	3,31	16,55	29,68	4,79	3,90	15,19	547%	-0,38	0,24
v	16078	16041	0	0,00%	-0,65	-7,63	-2,61	-0,86	1,08	12,62	20,25	3,69	2,49	6,20	-383%	0,45	-0,06
visibility	16078	10543	0	0,00%	21668	21,07	24038	24042	24055	24235	24214	17,48	7043	49606128	33%	-2,66	5,15
vlev1	16078	16049	0	0,00%	-0,90	-9,94	-3,56	-1,29	1,56	16,62	26,56	5,12	3,32	11,04	-369%	0,47	-0,19
vlev2	16078	16052	0	0,00%	-1,24	-11,55	-4,72	-1,76	1,92	19,48	31,03	6,64	4,17	17,42	-335%	0,47	-0,39
vlev3	16078	16055	0	0,00%	-1,46	-12,48	-5,32	-2,02	2,04	20,93	33,40	7,36	4,68	21,89	-321%	0,46	-0,44
weasd	16078	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
wind_gust	16078	16072	0	0,00%	5,13	0,01	2,94	4,47	6,85	23,17	23,16	3,92	3,02	9,15	59%	0,99	1,08
y	16078	7500	0	0,00%	2,46	0,00	0,00	0,02	4,66	11,69	11,69	4,66	3,54	12,56	144%	1,19	-0,08

Table 6.3: Univariate analysis for site 2.

Variable	Counters				Location								Dispersion			Shape	
	Count	Count Distinct	Missing Values [#]	Missing Values [%]	Mean	Min	Q1	Median	Q3	Max	Range	IQR	Standard Deviation	Variance	Variation Coefficient	Skewness	Kurtosis
cape	16078	3573	0	0,00%	50,59	0,00	0,00	0,00	5,97	2140	2140	5,97	165,51	27393	327%	4,98	30,23
cfh	16054	213	24	0,15%	0,08	0,00	0,00	0,00	0,10	1,00	1,00	0,10	0,17	0,03	204%	2,75	8,68
cfl	16054	3168	24	0,15%	0,10	0,00	0,00	0,00	0,12	1,00	1,00	0,12	0,19	0,04	186%	1,81	2,20
cfm	16054	181	24	0,15%	0,03	0,00	0,00	0,00	0,00	1,00	1,00	0,00	0,10	0,01	356%	4,86	28,03
cft	16054	2717	24	0,15%	0,17	0,00	0,00	0,03	0,30	1,00	1,00	0,30	0,23	0,05	133%	1,31	1,06
cin	16078	4615	0	0,00%	-20,09	-1326,05	-0,04	0,00	0,00	0,05	1326	0,04	84,99	7223	-423%	-6,22	48,62
conv_prec	16054	1869	24	0,15%	0,02	0,00	0,00	0,00	0,00	5,20	5,20	0,00	0,18	0,03	899%	12,95	210,37
dir	16078	16059	0	0,00%	223,37	0,01	138,50	253,69	313,72	359,97	359,96	175,22	104,04	10824	47%	-0,56	-0,95
HGT500	16078	15716	0	0,00%	5690	5247	5620	5710	5777	5924	677,18	157,34	114,39	13084	2,01%	-0,71	0,05
HGT850	16054	15287	24	0,15%	1507	1333	1483	1513	1537	1621	287,76	54,09	44,06	1941	2,92%	-0,65	0,61
HGTlev1	16078	14837	0	0,00%	247,40	245,70	247,02	247,36	247,75	249,16	3,46	0,73	0,56	0,31	0,23%	0,26	-0,08
HGTlev2	16078	15114	0	0,00%	289,94	285,45	288,95	289,82	290,87	294,61	9,16	1,92	1,47	2,16	0,51%	0,26	-0,07
HGTlev3	16078	15393	0	0,00%	324,15	317,48	322,67	323,97	325,53	331,09	13,61	2,86	2,17	4,71	0,67%	0,26	-0,06
land_use	16054	1	24	0,15%	2,00	2,00	2,00	2,00	2,00	2,00	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
lhflx	16054	14361	24	0,15%	61,99	-13,36	1,82	14,05	102,39	401,30	414,66	100,56	86,44	7471	139%	1,41	0,89
lwflx	16054	16029	24	0,15%	325,32	213,31	297,41	324,60	354,44	438,24	224,93	57,03	38,47	1480	12%	-0,03	-0,59
lwm	16078	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
meteograms	16054	13	24	0,15%	102,27	101,00	101,00	101,00	103,00	119,00	18,00	2,00	3,10	9,64	3,04%	4,08	17,46
mod	16054	16048	24	0,15%	3,19	0,03	1,77	2,93	4,28	14,68	14,64	2,50	1,84	3,39	58%	0,85	0,80
mslp	16078	15415	0	0,00%	101857	99869	101510	101828	102201	103478	3609	690,87	550,87	303462	0,54%	0,04	0,26
pbl_height	16078	16070	0	0,00%	400,02	23,97	99,92	255,68	578,52	2288	2264	478,60	389,98	152088	97%	1,36	1,48
prec	16054	1784	24	0,15%	0,04	-0,04	0,00	0,00	0,00	11,29	11,33	0,00	0,33	0,11	744%	15,89	361,83
rh	16078	15074	0	0,00%	0,82	0,30	0,70	0,88	0,97	1,00	0,70	0,27	0,17	0,03	21%	-0,88	-0,38
shflx	16078	16031	0	0,00%	42,55	-213,18	-12,28	-4,69	69,64	424,16	637,34	81,92	93,73	8785	220%	1,63	1,81
snow_prec	16078	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
snowlevel	16078	16070	0	0,00%	2487	446,12	1957	2438	3029	4299	3853	1071	705,68	497988	28%	0,04	-0,67
sst	16078	15738	0	0,00%	289,13	270,40	284,22	288,16	293,36	310,63	40,23	9,14	7,10	50,46	2,46%	0,45	-0,15
swflx	16078	6457	0	0,00%	210,33	0,00	0,00	0,00	383,97	1033	1033	383,97	309,34	95691	147%	1,26	0,17
T500	16078	15849	0	0,00%	259,26	241,84	255,23	259,74	263,40	271,31	29,47	8,17	5,29	27,95	2,04%	-0,30	-0,50
T850	16078	15813	0	0,00%	284,42	268,15	280,15	283,87	288,64	301,45	33,29	8,49	5,64	31,86	1,98%	0,16	-0,50
temp	16054	15687	24	0,15%	288,77	271,01	284,45	288,09	292,53	308,45	37,44	8,08	6,35	40,28	2,20%	0,40	-0,06
topo	16054	1	24	0,15%	221,92	221,92	221,92	221,92	221,92	221,92	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
u	16054	16006	24	0,15%	0,78	-9,75	-0,63	0,92	2,16	11,15	20,90	2,79	2,41	5,82	309%	-0,14	0,58
ulev1	16078	16057	0	0,00%	0,96	-11,40	-0,90	1,28	2,79	13,01	24,41	3,69	2,98	8,90	312%	-0,24	0,25
ulev2	16078	16049	0	0,00%	1,10	-12,87	-1,15	1,51	3,46	14,93	27,80	4,60	3,63	13,18	329%	-0,31	0,08
ulev3	16078	16045	0	0,00%	1,11	-13,60	-1,34	1,55	3,73	15,81	29,41	5,07	4,00	15,96	361%	-0,33	0,07
v	16054	16016	24	0,15%	-0,39	-9,53	-2,10	-0,62	1,12	13,79	23,31	3,22	2,65	7,02	-676%	0,44	0,72
visibility	16078	10455	0	0,00%	21740	16,61	24038	24042	24056	24235	24218	17,88	7015	49215405	32%	-2,71	5,40
vlev1	16054	16031	24	0,15%	-0,47	-11,33	-2,71	-0,87	1,57	16,28	27,61	4,28	3,27	10,70	-690%	0,47	0,37
vlev2	16054	16030	24	0,15%	-0,97	-13,38	-3,90	-1,60	1,50	18,76	32,14	5,40	4,04	16,34	-415%	0,60	0,16
vlev3	16054	16030	24	0,15%	-1,28	-14,59	-4,54	-2,01	1,49	20,47	35,06	6,04	4,53	20,49	-353%	0,62	0,08
weasd	16078	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
wind_gust	16078	16072	0	0,00%	5,13	0,04	2,67	4,57	6,94	23,17	23,13	4,27	3,15	9,92	61%	0,98	1,04
y	16078	4893	0	0,00%	11,41	0,00	0,00	0,51	23,12	47,26	47,26	23,12	15,54	241,54	136%	1,02	-0,52

Table 6.4: Univariate analysis for site 3.

Variable	Counters				Location								Dispersion			Shape	
	Count	Count Distinct	Missing Values [#]	Missing Values [%]	Mean	Min	Q1	Median	Q3	Max	Range	IQR	Standard Deviation	Variance	Variation Coefficient	Skewness	Kurtosis
cape	16078	3468	0	0,00%	43,26	0,00	0,00	0,00	4,23	2052	2052	4,23	146,98	21603	340%	5,32	35,78
cfh	16078	214	0	0,00%	0,09	0,00	0,00	0,00	0,10	1,00	1,00	0,10	0,18	0,03	199%	2,65	7,96
cfl	16078	2666	0	0,00%	0,09	0,00	0,00	0,00	0,06	1,00	1,00	0,06	0,18	0,03	197%	1,91	2,52
cfm	16078	185	0	0,00%	0,03	0,00	0,00	0,00	0,00	1,00	1,00	0,00	0,10	0,01	350%	4,63	24,57
cft	16078	2286	0	0,00%	0,16	0,00	0,00	0,01	0,30	1,00	1,00	0,30	0,22	0,05	136%	1,39	1,39
cin	16078	4431	0	0,00%	-17,20	-1219,98	-0,03	0,00	0,00	0,05	1220	0,03	80,31	6450	-467%	-7,01	60,16
conv_prec	16078	1901	0	0,00%	0,03	-0,04	0,00	0,00	0,00	5,58	5,62	0,00	0,24	0,06	888%	12,23	178,88
dir	16078	16050	0	0,00%	231,02	0,03	146,95	273,98	322,61	360,00	359,97	175,66	107,03	11454	46%	-0,73	-0,82
HGT500	16078	15748	0	0,00%	5694	5257	5623	5714	5781	5926	669,11	157,96	114,90	13202	2,02%	-0,70	0,03
HGT850	16078	15433	0	0,00%	1508	1315	1480	1515	1543	1637	322,02	63,66	52,81	2788	3,50%	-0,72	0,55
HGTlev1	16078	15431	0	0,00%	124,21	122,58	123,84	124,18	124,55	125,88	3,30	0,71	0,55	0,30	0,44%	0,18	-0,08
HGTlev2	16078	15127	0	0,00%	166,78	162,49	165,81	166,71	167,69	171,17	8,68	1,88	1,44	2,07	0,86%	0,18	-0,07
HGTlev3	16078	15416	0	0,00%	201,00	194,64	199,58	200,89	202,36	207,53	12,89	2,78	2,13	4,53	1,06%	0,18	-0,06
land_use	16078	1	0	0,00%	6,00	6,00	6,00	6,00	6,00	6,00	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
lhflx	16078	14499	0	0,00%	67,38	-12,28	2,47	18,60	105,73	440,50	452,78	103,26	93,10	8668	138%	1,46	1,11
lwflx	16078	16045	0	0,00%	327,81	216,82	300,23	327,15	356,67	436,65	219,83	56,44	38,29	1466	12%	-0,03	-0,57
lwm	16078	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
meteograms	16078	13	0	0,00%	102,12	101,00	101,00	101,00	103,00	119,00	18,00	2,00	2,88	8,31	2,82%	4,41	20,93
mod	16078	16072	0	0,00%	2,91	0,01	1,60	2,72	3,97	12,61	12,60	2,37	1,64	2,70	57%	0,67	0,27
mslp	16078	15433	0	0,00%	101845	99885	101494	101817	102191	103493	3609	697,45	561,40	315172	0,55%	0,05	0,24
pbl_height	16078	16075	0	0,00%	387,15	23,91	108,72	255,17	557,60	2312	2288	448,88	365,66	133705	94%	1,35	1,48
prec	16078	1737	0	0,00%	0,05	-0,05	0,00	0,00	0,00	12,24	12,29	0,00	0,37	0,14	776%	15,13	316,39
rh	16078	14773	0	0,00%	0,83	0,32	0,71	0,89	0,97	1,00	0,68	0,26	0,17	0,03	20%	-0,88	-0,35
shflx	16078	16036	0	0,00%	40,09	-137,53	-16,79	-5,44	70,37	421,15	558,68	87,15	94,82	8991	236%	1,54	1,47
snow_prec	16078	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
snowlevel	16078	16069	0	0,00%	2446	422,21	1925	2402	2975	4231	3809	1050	697,68	486763	29%	0,04	-0,67
sst	16078	15717	0	0,00%	289,18	270,82	284,54	288,41	293,13	310,02	39,20	8,59	6,75	45,62	2,34%	0,41	-0,12
swflx	16078	6437	0	0,00%	207,34	0,00	0,00	0,00	373,28	1022	1022	373,28	306,84	94148	148%	1,27	0,19
T500	16078	15811	0	0,00%	259,13	241,33	255,02	259,57	263,31	271,18	29,85	8,29	5,30	28,14	2,05%	-0,29	-0,50
T850	16078	15839	0	0,00%	284,11	267,98	279,88	283,55	288,27	300,74	32,76	8,39	5,60	31,38	1,97%	0,17	-0,53
temp	16078	15711	0	0,00%	288,93	271,31	284,74	288,43	292,61	308,05	36,75	7,87	6,18	38,15	2,14%	0,33	-0,06
topo	16078	1	0	0,00%	98,72	98,72	98,72	98,72	98,72	98,72	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
u	16078	16030	0	0,00%	1,05	-8,92	-0,32	1,07	2,26	10,29	19,21	2,59	2,12	4,48	202%	0,07	0,43
ulev1	16078	16056	0	0,00%	1,36	-10,75	-0,46	1,51	3,06	12,74	23,49	3,52	2,68	7,16	196%	-0,05	0,14
ulev2	16078	16055	0	0,00%	1,47	-12,18	-0,81	1,66	3,78	14,77	26,95	4,59	3,34	11,16	228%	-0,11	-0,13
ulev3	16078	16049	0	0,00%	1,45	-12,76	-0,98	1,64	4,05	15,84	28,60	5,03	3,73	13,92	257%	-0,17	-0,12
v	16078	16039	0	0,00%	-0,85	-7,70	-2,38	-0,99	0,50	12,28	19,98	2,89	2,21	4,88	-261%	0,46	0,72
visibility	16078	10335	0	0,00%	21882	16,61	24038	24042	24056	24235	24218	17,69	6823	46559668	31%	-2,83	6,06
vlev1	16078	16051	0	0,00%	-1,12	-9,67	-3,22	-1,40	0,69	15,30	24,97	3,91	2,84	8,05	-254%	0,50	0,41
vlev2	16078	16049	0	0,00%	-1,47	-11,85	-4,25	-1,84	0,87	18,16	30,01	5,12	3,61	13,02	-245%	0,51	0,10
vlev3	16078	16058	0	0,00%	-1,68	-13,13	-4,83	-2,05	0,97	19,65	32,77	5,80	4,10	16,81	-244%	0,48	-0,04
weasd	16078	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
wind_gust	16078	16074	0	0,00%	4,80	0,02	2,52	4,18	6,54	22,33	22,32	4,02	3,02	9,10	63%	0,99	1,00
y	16078	7069	0	0,00%	1,23	0,00	0,00	0,05	2,46	4,96	4,96	2,46	1,70	2,89	138%	1,05	-0,46

Table 6.5: Univariate analysis for site 4.

Variable	Counters				Location								Dispersion			Shape	
	Count	Count Distinct	Missing Values [#]	Missing Values [%]	Mean	Min	Q1	Median	Q3	Max	Range	IQR	Standard Deviation	Variance	Variation Coefficient	Skewness	Kurtosis
cape	9383	2280	0	0,00%	49,03	0,00	0,00	0,00	4,83	1817	1817	4,83	163,30	26666	333%	4,94	28,79
cfh	9383	184	0	0,00%	0,09	0,00	0,00	0,00	0,10	1,00	1,00	0,10	0,18	0,03	197%	2,65	7,90
cfl	9383	1678	0	0,00%	0,10	0,00	0,00	0,00	0,10	1,00	1,00	0,10	0,19	0,04	190%	1,81	2,11
cfm	9383	142	0	0,00%	0,03	0,00	0,00	0,00	0,00	1,00	1,00	0,00	0,11	0,01	331%	4,26	20,40
cft	9383	1493	0	0,00%	0,17	0,00	0,00	0,03	0,30	1,00	1,00	0,30	0,23	0,05	133%	1,33	1,11
cin	9383	2850	0	0,00%	-16,98	-1230,04	-0,02	0,00	0,00	0,05	1230	0,03	79,30	6289	-467%	-7,27	65,89
conv_prec	9383	1180	0	0,00%	0,03	-0,01	0,00	0,00	0,00	4,08	4,09	0,00	0,21	0,04	793%	11,19	151,85
dir	9383	9373	0	0,00%	218,12	0,13	127,68	257,42	307,60	359,89	359,75	179,92	103,71	10755	48%	-0,53	-1,14
HGT500	9383	9269	0	0,00%	5680	5249	5610	5695	5768	5926	676,56	158,23	114,61	13136	2,02%	-0,66	0,17
HGT850	9383	9143	0	0,00%	1501	1322	1475	1508	1534	1629	307,41	58,85	48,88	2389	3,26%	-0,78	0,61
HGTlev1	9383	8966	0	0,00%	181,78	180,15	181,41	181,75	182,14	183,56	3,41	0,73	0,56	0,31	0,31%	0,11	-0,11
HGTlev2	9383	9072	0	0,00%	224,27	219,97	223,30	224,20	225,23	228,95	8,98	1,93	1,47	2,16	0,65%	0,12	-0,10
HGTlev3	9383	9145	0	0,00%	258,44	251,93	256,99	258,33	259,84	265,41	13,48	2,85	2,16	4,68	0,84%	0,14	-0,08
land_use	9383	1	0	0,00%	2,00	2,00	2,00	2,00	2,00	2,00	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
lhflx	9383	8808	0	0,00%	61,27	-14,06	3,72	18,41	94,34	394,70	408,76	90,62	83,28	6936	136%	1,48	1,15
lwflx	9383	9377	0	0,00%	324,15	216,11	296,58	323,28	352,95	426,00	209,90	56,37	38,51	1483	12%	-0,04	-0,52
lwm	9383	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
meteograms	9383	13	0	0,00%	102,59	101,00	101,00	101,00	103,00	119,00	18,00	2,00	3,80	14,47	3,71%	3,33	10,48
mod	9383	9383	0	0,00%	3,41	0,01	1,94	3,12	4,58	15,19	15,18	2,64	1,99	3,94	58%	0,86	0,97
mslp	9383	9164	0	0,00%	101832	99850	101546	101836	102170	103327	3477	623,84	522,94	273470	0,51%	-0,29	0,51
pbl_height	9383	9383	0	0,00%	397,90	24,02	103,16	252,43	569,51	2230	2206	466,35	388,01	150556	98%	1,39	1,50
prec	9383	1339	0	0,00%	0,06	-0,05	0,00	0,00	0,00	17,80	17,85	0,00	0,44	0,19	687%	18,70	522,43
rh	9383	9040	0	0,00%	0,81	0,31	0,70	0,87	0,96	1,00	0,69	0,25	0,17	0,03	21%	-0,88	-0,31
shflx	9383	9368	0	0,00%	37,40	-238,25	-15,04	-5,37	58,48	411,66	649,91	73,52	92,41	8540	247%	1,74	2,29
snow_prec	9383	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
snowlevel	9383	9380	0	0,00%	2397	409,58	1916	2362	2894	4267	3858	978,83	669,42	448122	28%	0,04	-0,41
sst	9383	9276	0	0,00%	288,99	271,13	284,31	288,18	293,20	310,70	39,56	8,89	6,91	47,72	2,39%	0,36	-0,15
swflx	9383	3841	0	0,00%	195,69	0,00	0,00	0,00	336,70	1024	1024	336,70	298,20	88923	152%	1,35	0,46
T500	9383	9294	0	0,00%	258,69	240,99	254,60	259,21	262,79	270,59	29,60	8,19	5,14	26,46	1,99%	-0,27	-0,43
T850	9383	9297	0	0,00%	283,67	267,66	279,77	283,22	287,60	301,12	33,46	7,83	5,49	30,17	1,94%	0,15	-0,12
temp	9383	9255	0	0,00%	288,64	271,73	284,53	288,08	292,46	308,54	36,80	7,94	6,22	38,71	2,16%	0,28	-0,07
topo	9383	1	0	0,00%	156,32	156,32	156,32	156,32	156,32	156,32	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
u	9383	9366	0	0,00%	0,69	-10,25	-1,18	1,03	2,60	10,87	21,12	3,78	2,92	8,53	424%	-0,23	0,08
ulev1	9383	9378	0	0,00%	0,83	-11,98	-1,66	1,43	3,21	12,79	24,78	4,88	3,58	12,79	432%	-0,27	-0,12
ulev2	9383	9374	0	0,00%	0,97	-13,65	-1,87	1,70	3,84	14,93	28,58	5,71	4,24	17,95	435%	-0,33	-0,19
ulev3	9383	9372	0	0,00%	1,03	-14,69	-1,97	1,75	4,12	16,09	30,78	6,09	4,58	20,97	445%	-0,35	-0,13
v	9383	9367	0	0,00%	-0,38	-8,96	-1,96	-0,72	0,89	13,82	22,78	2,85	2,54	6,45	-663%	0,74	1,22
visibility	9383	6947	0	0,00%	22058	30,47	24038	24042	24054	24235	24205	16,28	6495	42185339	29%	-3,00	7,12
vlev1	9383	9372	0	0,00%	-0,50	-10,50	-2,53	-0,97	1,18	16,40	26,89	3,71	3,10	9,62	-619%	0,73	0,91
vlev2	9383	9375	0	0,00%	-0,81	-12,84	-3,49	-1,49	1,42	19,97	32,81	4,91	3,81	14,54	-469%	0,77	0,63
vlev3	9383	9375	0	0,00%	-1,02	-13,56	-4,12	-1,79	1,59	21,56	35,11	5,71	4,26	18,16	-418%	0,77	0,45
weasd	9383	1	0	0,00%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
wind_gust	9383	9378	0	0,00%	5,08	0,02	2,72	4,45	6,78	24,77	24,76	4,06	3,22	10,40	63%	1,14	1,57
y	9383	4159	0	0,00%	8,38	0,00	0,00	0,18	16,27	36,17	36,17	16,27	11,98	143,61	143%	1,15	-0,22

Table 6.6: Univariate analysis for site 5.

Variable			Counters						Location					Dispersion			Shape	
	Count	Count Distinct	Missing Values [#]	Missing Values [%]	Mean	Min	Q1	Median	Q3	Max	Range	IQR	Standard Deviation	Variance	Variation Coefficient	Skewness	Kurtosis
cape	33140	6910	144	0,43%	52,69	0,00	0,00	0,00	16,40	2669	2669	16,40	173,34	30046	329%	5,83	43,64
cfh	33140	266	144	0,43%	0,08	0,00	0,00	0,00	0,11	1,00	1,00	0,11	0,16	0,03	190%	2,62	8,34
cfl	33140	5308	144	0,43%	0,19	0,00	0,00	0,00	0,44	1,00	1,00	0,44	0,27	0,08	145%	1,08	-0,28
cfm	33140	218	144	0,43%	0,06	0,00	0,00	0,00	0,00	1,00	1,00	0,00	0,16	0,03	272%	3,21	10,29
cft	33140	4952	144	0,43%	0,25	0,00	0,00	0,13	0,47	1,00	1,00	0,47	0,28	0,08	113%	0,80	-0,57
cin	33140	7864	144	0,43%	-12,22	-909,36	-0,01	0,00	0,00	0,05	909,41	0,01	53,52	2865	-438%	-6,50	53,13
conv_prec	33140	4123	144	0,43%	0,06	-0,02	0,00	0,00	0,00	9,03	9,05	0,00	0,30	0,09	534%	7,69	85,08
dir	33140	33079	144	0,43%	185,69	0,00	75,44	195,73	293,03	360,00	360,00	217,59	113,71	12930	61%	-0,04	-1,44
HGT500	33140	31833	144	0,43%	5645	5189	5572	5662	5738	5898	708,86	166,15	120,39	14495	2,13%	-0,68	0,00
HGT850	33140	30911	144	0,43%	1529	1125	1497	1539	1572	1672	546,94	75,25	64,77	4195	4,24%	-1,04	1,79
HGTlev1	33140	20533	144	0,43%	541,21	539,76	540,79	541,14	541,59	543,25	3,50	0,79	0,56	0,31	0,10%	0,41	-0,29
HGTlev2	33140	27116	144	0,43%	583,30	579,44	582,19	583,12	584,32	588,69	9,25	2,13	1,49	2,22	0,26%	0,40	-0,32
HGTlev3	33140	28895	144	0,43%	617,15	611,34	615,47	616,86	618,68	625,19	13,85	3,21	2,23	4,98	0,36%	0,39	-0,36
land_use	33140	1	144	0,43%	14,00	14,00	14,00	14,00	14,00	14,00	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
lhflx	33140	30016	144	0,43%	62,27	-10,86	4,83	23,29	91,17	457,33	468,19	86,35	81,69	6674	131%	1,58	1,69
lwflx	33140	33019	144	0,43%	318,26	203,63	285,70	321,58	352,25	420,11	216,48	66,55	41,55	1727	13%	-0,24	-0,84
lwm	33140	1	144	0,43%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
meteograms	33140	16	144	0,43%	103,35	101,00	101,00	101,00	104,00	120,00	19,00	3,00	4,17	17,41	4,04%	2,33	5,01
mod	33140	33111	144	0,43%	3,47	0,02	2,14	3,24	4,52	15,39	15,37	2,38	1,83	3,34	53%	0,87	1,20
mslp	33140	30644	144	0,43%	101868	97498	101515	101852	102253	103716	6218	737,98	675,09	455752	0,66%	-0,41	1,56
pbl_height	33140	33125	144	0,43%	394,40	23,96	93,79	249,51	582,04	2682	2658	488,25	387,73	150332	98%	1,41	1,82
prec	33140	4663	144	0,43%	0,17	-0,10	0,00	0,00	0,01	18,73	18,83	0,01	0,70	0,48	410%	8,37	101,18
rh	33138	31588	146	0,44%	0,82	0,28	0,70	0,86	0,96	1,00	0,72	0,26	0,17	0,03	20%	-0,81	-0,36
shflx	33140	32993	144	0,43%	38,25	-183,90	-25,02	-3,98	62,53	527,62	711,52	87,54	114,14	13028	298%	1,64	2,19
snow_prec	33140	8	144	0,43%	0,00	0,00	0,00	0,00	0,00	0,00	0,01	0,00	0,00	0,00	-2759332868484%	-1,71	14271
snowlevel	33140	33106	144	0,43%	2166	83,02	1619	2118	2757	4072	3989	1138	765,14	585436	35%	0,04	-0,63
sst	33140	31838	144	0,43%	286,34	270,48	281,70	285,24	290,02	310,45	39,98	8,32	6,52	42,56	2,28%	0,63	0,03
swflx	33140	11356	144	0,43%	186,60	0,00	0,00	0,00	317,92	1025	1025	317,92	289,70	83926	155%	1,40	0,61
T500	33140	32148	144	0,43%	257,90	238,29	254,29	258,19	262,01	270,04	31,75	7,72	5,36	28,73	2,08%	-0,39	-0,14
T850	33140	32117	144	0,43%	282,48	267,98	278,16	282,01	286,77	300,46	32,48	8,60	5,78	33,42	2,05%	0,21	-0,62
temp	33140	31806	144	0,43%	286,29	271,00	281,82	285,34	289,93	309,58	38,58	8,11	6,20	38,48	2,17%	0,59	-0,01
topo	33140	1	144	0,43%	516,00	516,00	516,00	516,00	516,00	516,00	0,00	0,00	0,00	0,00	0,00%	0,00	0,00
u	33140	32999	144	0,43%	0,32	-10,14	-1,58	0,21	2,26	12,75	22,89	3,83	2,84	8,04	874%	0,14	0,16
ulev1	33140	33035	144	0,43%	0,30	-13,29	-2,46	0,30	2,93	16,61	29,90	5,38	3,78	14,29	1257%	0,13	-0,02
ulev2	33140	33065	144	0,43%	0,27	-15,95	-2,96	0,56	3,48	19,77	35,72	6,45	4,65	21,61	1717%	0,02	-0,05
ulev3	33140	33074	144	0,43%	0,26	-17,36	-3,08	0,76	3,75	21,34	38,70	6,83	5,12	26,23	1935%	-0,08	0,00
v	33140	32981	144	0,43%	-0,45	-9,61	-2,14	-0,79	1,00	13,05	22,66	3,14	2,65	7,02	-583%	0,57	0,71
visibility	33140	19085	144	0,43%	21075	11,23	24038	24045	24215	24235	24224	176,70	7665	58752981	36%	-2,23	3,13
vlev1	33140	33034	144	0,43%	-0,71	-12,39	-3,15	-1,16	1,35	17,06	29,46	4,50	3,54	12,52	-500%	0,58	0,53
vlev2	33140	33037	144	0,43%	-0,73	-14,05	-3,74	-1,12	1,80	20,11	34,16	5,54	4,35	18,93	-597%	0,50	0,29
vlev3	33140	33057	144	0,43%	-0,70	-14,71	-4,02	-1,08	2,13	21,59	36,30	6,15	4,80	23,03	-687%	0,45	0,17
weasd	33139	1	145	0,44%	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00		0,00	0,00
wind_gust	33140	33114	144	0,43%	5,89	0,04	3,34	4,94	7,74	27,04	27,00	4,40	3,63	13,18	62%	1,14	1,25
y	33284	11737	0	0,00%	5,92	0,00	0,00	0,00	8,80	5927	5927	8,80	36,97	1366	624%	131,84	20215

Table 6.7: Univariate analysis for site 1023.

C Bivariate Analysis

C.1 Correlation Matrix

Calculated with Pearson's correlation coefficient for numerical variables only.

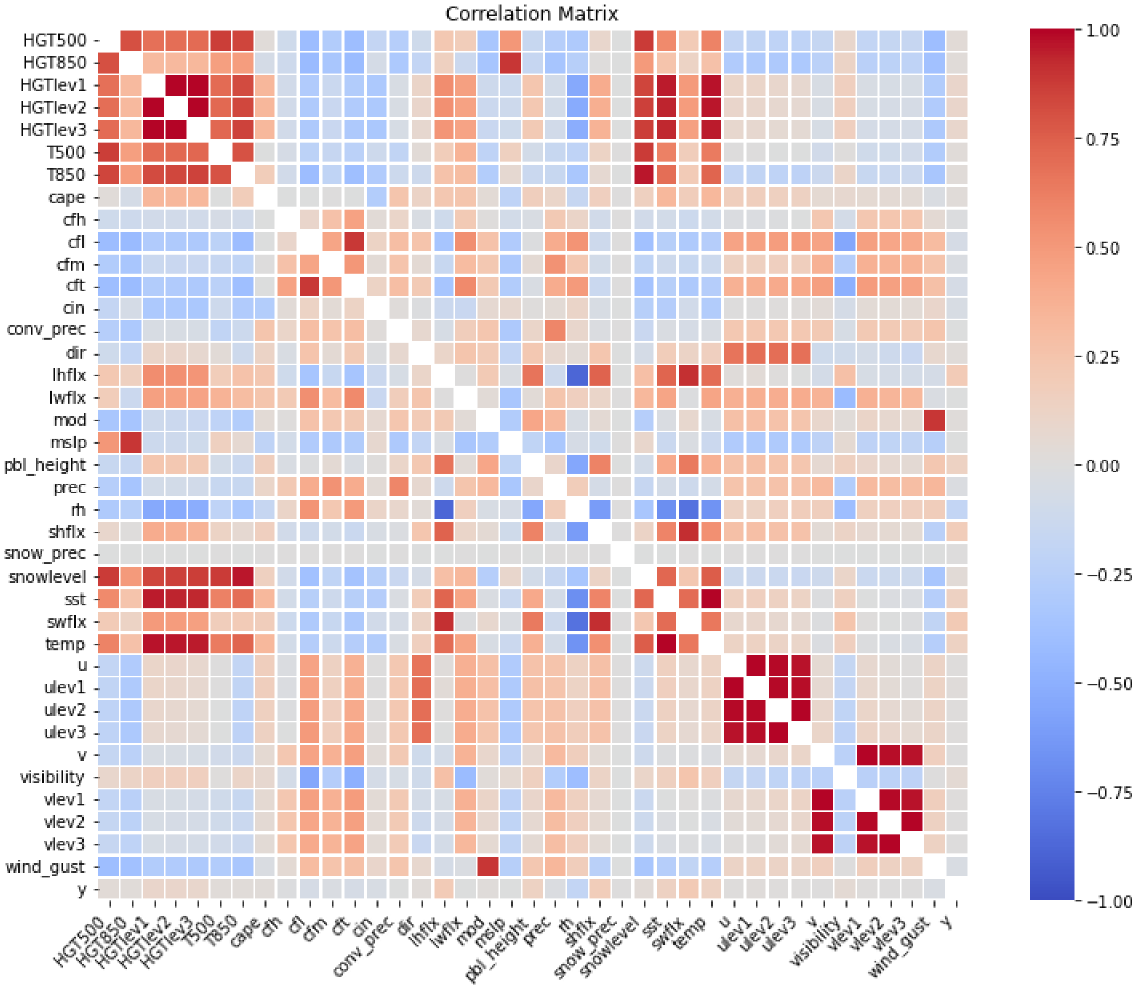


Figure 6.1: Correlation matrix for site 1023 dataset.

C.2 Correlation between Pairs of Variables

Var1	Var2	Correlation
HGTlev1	HGTlev2	99,92%
HGTlev2	HGTlev1	99,92%
HGTlev3	HGTlev2	99,90%
HGTlev2	HGTlev3	99,90%
sst	temp	99,73%
temp	sst	99,73%
HGTlev3	HGTlev1	99,69%
HGTlev1	HGTlev3	99,69%
ulev1	u	99,67%
u	ulev1	99,67%
ulev2	ulev3	99,63%
ulev3	ulev2	99,63%
v	vlev1	99,58%
vlev1	v	99,58%
vlev3	vlev2	99,51%
vlev2	vlev3	99,51%
ulev1	ulev2	99,21%
ulev2	ulev1	99,21%
vlev2	vlev1	98,73%
vlev1	vlev2	98,73%
ulev2	u	98,51%
u	ulev2	98,51%
v	vlev2	98,38%
vlev2	v	98,38%
ulev3	ulev1	98,06%
ulev1	ulev3	98,06%
ulev3	u	97,30%
u	ulev3	97,30%
vlev3	vlev1	97,24%
vlev1	vlev3	97,24%
HGTlev1	temp	97,09%
temp	HGTlev1	97,09%
vlev3	v	97,00%
v	vlev3	97,00%
snowlevel	T850	96,51%
T850	snowlevel	96,51%
temp	HGTlev2	96,45%
HGTlev2	temp	96,45%
temp	HGTlev3	95,45%
HGTlev3	temp	95,45%
HGTlev1	sst	95,33%
sst	HGTlev1	95,33%
HGTlev2	sst	94,53%
sst	HGTlev2	94,53%
sst	HGTlev3	93,33%
HGTlev3	sst	93,33%
shflx	swflx	91,83%
swflx	shflx	91,83%
lhflx	swflx	90,81%
swflx	lhflx	90,81%
mslp	HGT850	89,52%
HGT850	mslp	89,52%
mod	wind_gust	89,08%
wind_gust	mod	89,08%
cfl	cft	88,66%
cft	cfl	88,66%
lhflx	rh	-88,08%
rh	lhflx	-88,08%
T500	snowlevel	87,91%
snowlevel	T500	87,91%
snowlevel	HGT500	87,88%
HGT500	snowlevel	87,88%
snowlevel	HGTlev3	87,22%
HGTlev3	snowlevel	87,22%
HGT500	T500	87,02%
T500	HGT500	87,02%
snowlevel	HGTlev2	85,97%
HGTlev2	snowlevel	85,97%
HGTlev3	T850	85,36%
T850	HGTlev3	85,36%
snowlevel	HGTlev1	84,95%
HGTlev1	snowlevel	84,95%
T850	HGT500	84,62%
HGT500	T850	84,62%
T850	HGTlev2	83,87%
HGTlev2	T850	83,87%
T850	HGTlev1	82,68%
HGTlev1	T850	82,68%
rh	swflx	-82,32%
swflx	rh	-82,32%
HGT500	HGT850	81,09%
HGT850	HGT500	81,09%

Table 6.8: Correlations above 80% between pairs of variables for site 1023.

C.3 Correlation with Target Variable

Variable	Correlation	Description
swflx	20,70%	Surface downwelling shortwave flux.
lhflx	20,15%	Surface downward latent heat flux.
rh	-18,57%	Relative humidity at 2m.
shflx	18,27%	Surface downward sensible heat flux.
sst	14,49%	Sea surface temperature.
pbl_height	13,93%	PBL Height.
temp	13,57%	Temperature at 2m.
HGTlev1	10,02%	Geopotential height at model level 1.
HGTlev2	9,70%	Geopotential height at model level 2.
HGTlev3	9,23%	Geopotential height at model level 3.
cfl	-6,20%	Cloud cover at low levels.
cft	-5,55%	Cloud cover at low and mid levels.
visibility	5,05%	Visibility.
cin	-3,91%	Convective inhibition.
wind_gust	-3,72%	Wind gust.
snowlevel	3,41%	Snow level.
HGT500	3,22%	Geopotential height at 500mb.
T850	2,98%	Temperature at 850mb.
cape	2,87%	Convective available potential energy.
HGT850	2,57%	Geopotential height at 850mb.
dir	2,36%	Wind direction at 10m.
cfm	-2,36%	Cloud cover at mid levels.
T500	2,35%	Temperature at 500mb.
prec	-2,29%	Total accumulated rainfall between each model output.
mod	2,23%	Wind module at 10m.
ulev1	1,80%	Lon-wind at model level 1.
ulev2	1,53%	Lon-wind at model level 2.
cfh	-1,50%	Cloud cover at high levels.
u	1,40%	Lon-wind at 10m.
ulev3	1,28%	Lon-wind at model level 3.
vlev1	0,79%	Lat-wind at model level 1.
conv_prec	-0,62%	Total accumulated convective rainfall between each model output.
vlev2	0,35%	Lat-wind at model level 2.
lwflx	-0,28%	Surface downwelling longwave flux.
mslp	-0,28%	Mean sea level pressure.
v	0,25%	Lat-wind at 10m.
vlev3	0,11%	Lat-wind at model level 3.
snow_prec	0,00%	Total accumulated large scale snowfall between each model output.

Table 6.9: Correlations with target variable for site 1023.

D Feature Importance according to XGBoost

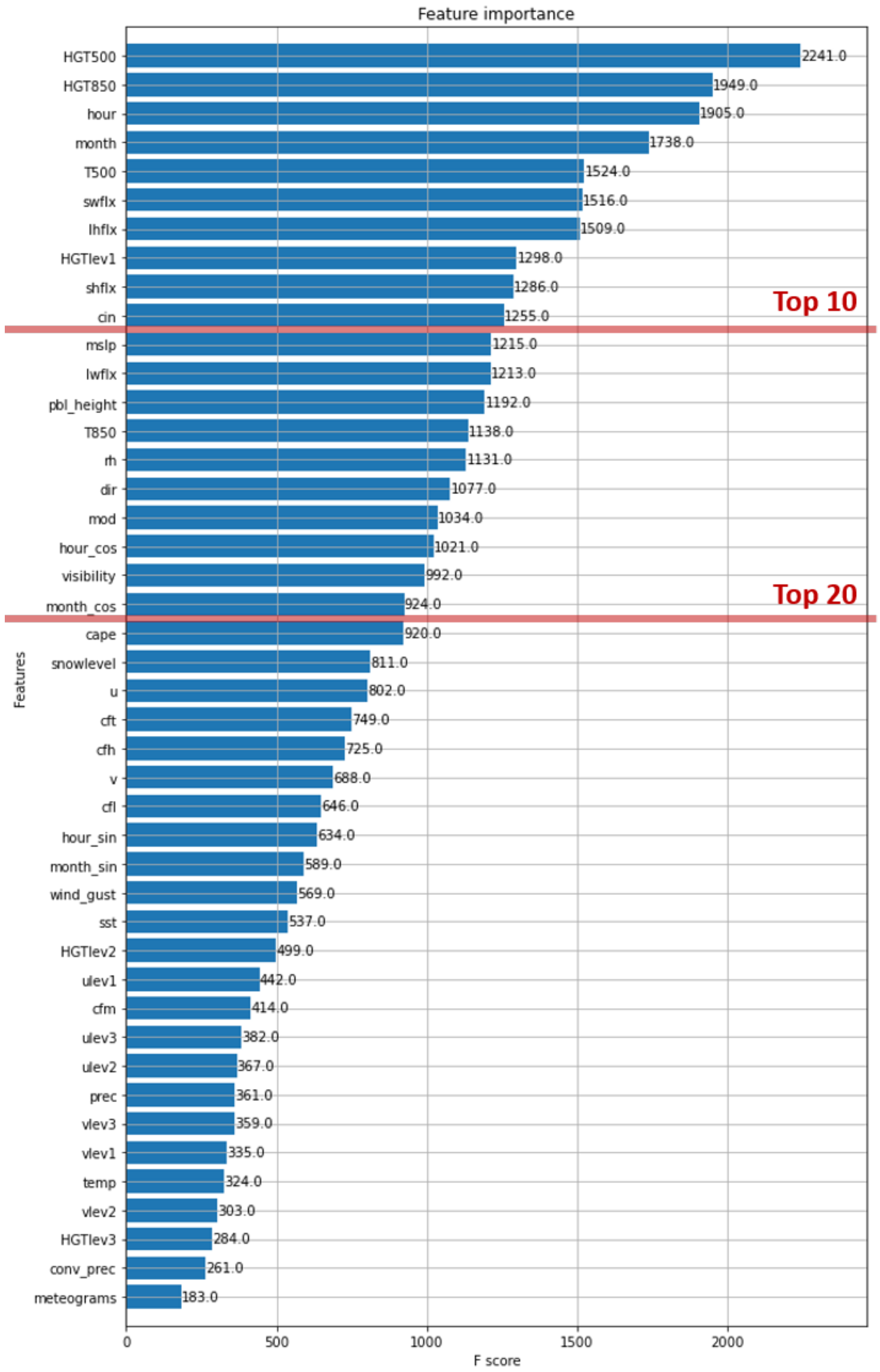


Figure 6.2: Feature importance, for site 1, including time predictors.