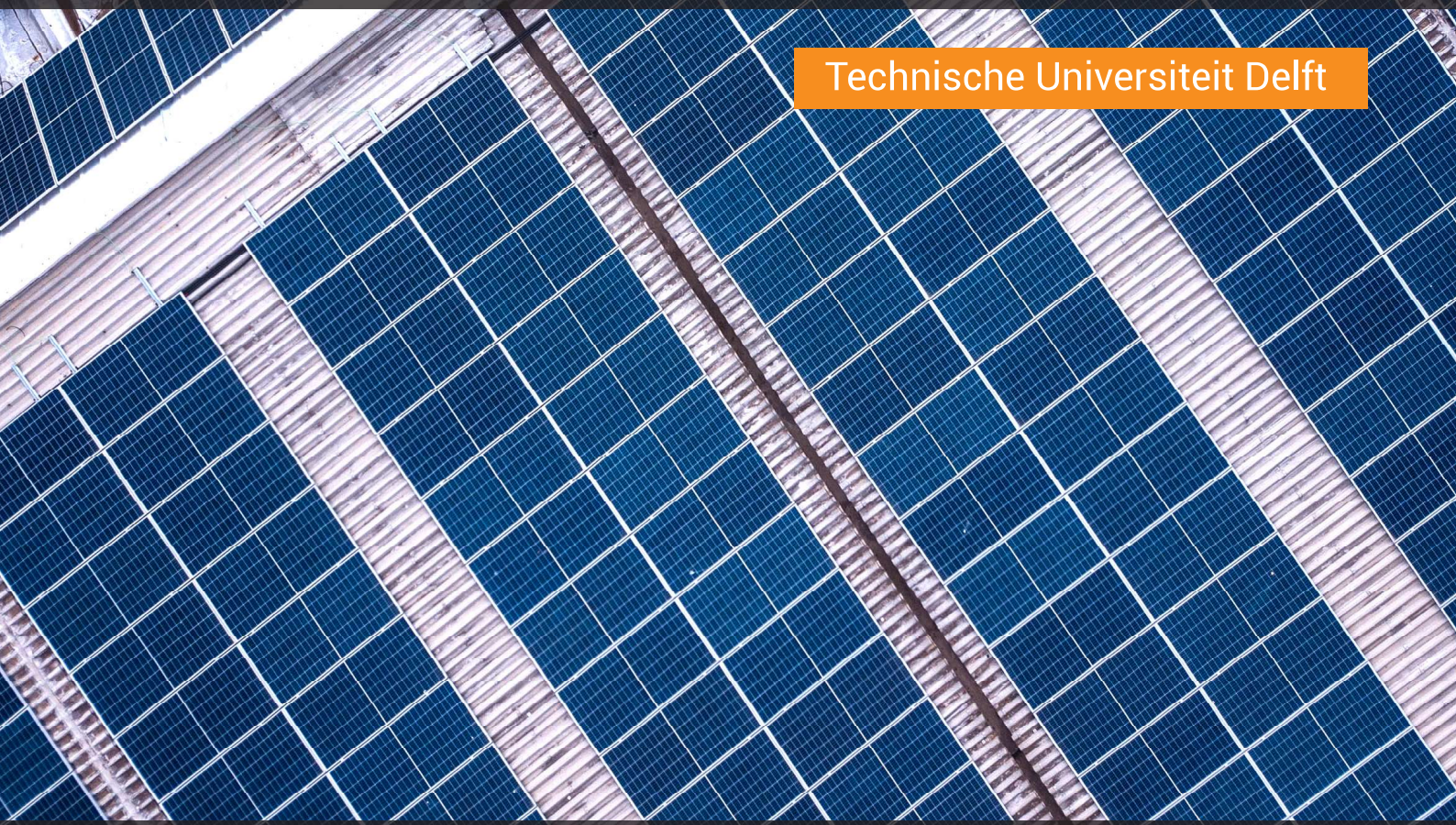
Photovoltaic Yield Nowcasting

For Residential Solar Systems in the Netherlands Using a Machine Learning Approach

D. Grzebyk

Technische Universiteit Delft



TU Delft

PHOTOVOLTAIC YIELD NOWCASTING FOR RESIDENTIAL SOLAR SYSTEMS IN THE NETHERLANDS USING A MACHINE LEARNING APPROACH

A thesis submitted to the Delft University of Technology in partial fulfillment of the requirements for the degree of

Master of Science in Sustainable Energy Technologies

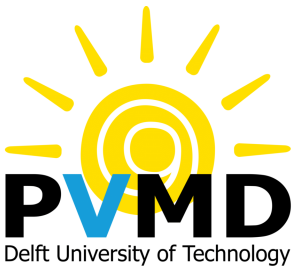
by

Daniel Grzebyk

July 2020

ISBN 999–99–9999–999–9

The work in this thesis was made in the:



PVMD Group
Department of Electrical Engineering, Mathematics and Computer Science
Faculty of Energy Transition
Delft University of Technology

Supervisors:	Dr. Olindo Isabella
	Dr. Hesan Ziar
	Jaap Donker
Co-reader:	Prof. dr. Marco Loog

ABSTRACT

An increasing number of photovoltaic (PV) systems are being installed worldwide and residential sector is responsible for a large part of this growth. Small scale PV systems do not have complex measuring devices and their breakdowns are not spotted immediately by the system owners. This might lead to prolonged time without generating power and creating both financial loss and environmental damage. This thesis presents a method of PV yield nowcasting laying foundations for remote monitoring. Early detection of faults is the first step towards eliminating the described issues. In this project four machine learning models for predicting solar yield were developed: ElasticNet, Polynomial Regression, Random Forest, and Extreme Gradient Boosting (XGBoost). The models were created both for daily and hourly data sets, as some inverters can log daily yields only. In both cases, the utilized data set consisted of data for the time between July 1st, 2018 and June 30th, 2019 and corresponding to 1,102 PV systems which is five times more than the largest data set studied in the found literature. The average PV system size in the data set is 4.44 kWp. Utilized inputs next to weather data and previous yields included shading factor describing fraction of direct light unable to reach PV system due to surrounding obstacles. Calculation of shading facor was based on 360° pictures taken at the site. XGBoost algorithm turned out to be the most suitable for the task of PV yield nowcasting obtaining Root Mean Squared Error (RMSE) of 1.48 kWh and Mean Absolute Error (MAE) of 0.877 kWh for hourly data aggregated to daily values and evaluated on future time steps. Currently used commercial software of Solar Monkey has RMSE equal 2.237 kWh and MAE equal 1.5 kWh. XGBoost model trained on daily data obtained RMSE 1.185 kWh and MAE 0.698 kWh outperforming hourly model most likely due to utilization of Hidden Markov Model for data cleaning. Next to overall performance, per system metrics were calculated for the hourly XGBoost. Mean individual RMSE for previously seen systems is 1.656 kWh while for unseen systems it equals 1.666 kWh. This means the model scales well to previously unseen systems and implies that its parallelized version is not necessary. Also, the model's learning saturates after seeing data corresponding to one year and 278 PV systems. Precalculation of GPOA worsened performance with respect to the model utilizing GHI. Hourly XGBoost has hourly RMSE of 0.281 kWh under clear sky and 0.377 kWh under partly cloudy sky which indicates it is more mistaken for cloudy conditions. This could be caused by low quality of cloud coverage data. The model also has large relative errors for small irradiance values which occur mostly in January and December, as well as just after sunrise and just before sunset. This issue is caused by using squared error as loss function during model training. Despite these shortcomings, the conclusive results recommend industrial implementation of the developed model.

Keywords: AI, artificial intelligence, forecasting, machine learning, nowcasting, photovoltaics, PV monitoring, renewable energy, solar yield prediction, yield nowcasting, XGBoost

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to dr Olindo Isabella and dr Hesan Ziar, my research supervisors, for their valuable suggestions and useful critiques of this research work. I appreciate very much they were willing to give their time so generously. I would like to thank also professor David Tax who provided me with very valuable information about the practical aspects of machine learning.

I would like to express my great appreciation to Jaap Donker, my company supervisor, for his patient guidance, enthusiastic encouragement and constructive recommendations on this project. His remarks changed my worldview and helped me grow as an engineer. I would like also to extend my thanks to Yitzi Snow and Rohi Perlsteyn, researchers in Solar Monkey, for their programming support and valuable discussions. Special thanks should be given to Yongming Luo, a Solar Monkey developer, for offering me the resources to run my models and his patience for my mistakes. I would like also to thank all Solar Monkey team for making my time at the company very enjoyable.

I wish to acknowledge the support of my friends. I would like to express my deep gratitude to Omkar Sane for our invaluable discussions about machine learning, renewable energy and welfare inequality which I used to write different parts of this work. I would like also to thank Sanket Mane for designing the cover of this thesis.

Finally, I wish to thank my family, especially my parents, for their enormous support and encouragement throughout my studies. I am extremely grateful they made possible everything that happened in the past two years.

CONTENTS

1 INTRODUCTION 3
1.1 Literature Study . . . 4
1.2 Research Objectives . . . 10
1.3 Solar Monkey . . . 10
1.4 Outline . . . 10
2 DATA 13
2.1 Data Quality . . . 13
2.1.1 Daily Inputs . . . 14
2.1.2 Hourly Inputs . . . 15
2.2 Feature Engineering . . . 17
2.2.1 Daily Inputs . . . 18
2.2.2 Hourly Rough Inputs . . . 19
2.2.3 Hourly Detailed Inputs . . . 19
2.3 Exploratory Data Analysis . . . 21
2.3.1 Daily Inputs . . . 24
2.3.2 Hourly Inputs . . . 24
2.3.3 Future Time Steps . . . 25
2.4 Data Scaling . . . 26
2.5 Dimensionality Reduction . . . 26
2.6 Feature Selection . . . 26
3 MACHINE LEARNING KEY CONCEPTS 29
3.1 Data Splitting . . . 29
3.2 Cross-Validation . . . 29
3.3 Cost functions and learning rates . . . 31
3.4 Bias vs. Variance Trade-off . . . 31
3.5 Hyperparameter Tuning . . . 33
3.6 Metrics . . . 34
4 RESULTS 39
4.1 Daily Data Set . . . 39
4.2 Hourly Rough Data Set . . . 40
4.3 Hourly Detailed Data Set . . . 41
4.4 Cross Data Set Analysis . . . 41
4.5 Feature Importance . . . 42
4.6 Learning Curves . . . 43
4.7 Identification of Incorrect String Configurations . . . 45
5 ADVANCED ANALYSIS 47
5.1 Previous Model . . . 47
5.2 Group Shuffle Split . . . 48
5.2.1 Past time steps . . . 49
5.2.2 Future Time steps . . . 50
5.3 Weather Analysis . . . 51
5.4 Error Analysis . . . 54
5.5 Weight Normalization . . . 55
6 CONCLUSIONS AND DISCUSSION 59
7 RECOMMENDATIONS 65
7.1 Solar Monkey . . . 65
7.2 Further Research . . . 67
A LITERATURE COMPARISON 77
B FEATURE IMPORTANCE 81
B.1 Daily Data Set . . . 81
B.2 Hourly Rough Data Set . . . 82

B.3 Hourly Detailed Data Set . . . 82
C OPTIMAL HYPERPARAMETER CONFIGURATIONS 85
C.1 Daily Data Set . . . 85
C.2 Hourly Rough Data Set . . . 86
C.3 Hourly Detailed Data Set . . . 86
D LEARNING CURVES 89
D.1 Daily Data . . . 89
D.2 Hourly Data . . . 92
E WORKING PRINCIPLES OF SELECTED MACHINE LEARNING MODELS 95
E.1 Persistence Method . . . 95
E.2 Elastic Net . . . 95
E.2.1 Ridge Regression . . . 95
E.2.2 Lasso Regression . . . 96
E.3 Polynomial Regression . . . 96
E.4 Support Vector Regression . . . 96
E.5 Random Forest . . . 97
E.6 Extreme Gradient Boosting . . . 98
E.7 Gradient Boosting with Quantile Loss Function . . . 98
E.8 Long Short-Term Memory Neural Network . . . 99
F EXPERIMENTAL SETUP 103
G SCIENTIFIC ARTICLE 105

LIST OF FIGURES

Figure 1.1 Classification of forecasting techniques [Raza et al., 2016] . . . 7
Figure 2.1 Types of data corruption [Solar Monkey, 2020] . . . 14
Figure 2.2 Fraction of corrupt data . . . 15
Figure 2.3 Example of constant yield . . . 16
Figure 2.4 Example of missing yield . . . 16
Figure 2.5 Example of high yield outliers . . . 16
Figure 2.6 Heatmap showing correlations between all features - daily data set . . . 18
Figure 2.7 Heatmap showing correlations between all features - hourly data set . . . 20
Figure 2.8 Flowchart of Global Plane of Array (GPOA) calculation - hourly detailed data set . . . 20
Figure 2.9 Flowchart of obstacle factor calculation - hourly detailed data set . . . 21
Figure 2.10 Module type histogram . . . 22
Figure 2.11 System size histogram . . . 22
Figure 2.12 System age histogram . . . 22
Figure 2.13 System age vs. system size . . . 23
Figure 2.14 Wind speed histogram - daily data set . . . 23
Figure 2.15 Ambient temperature histogram for hourly data . . . 23
Figure 2.16 Global Horizontal Irradiance (GHI) histogram - daily data set 24
Figure 2.17 Cloud coverage histogram - daily data . . . 24
Figure 2.18 GHI histogram - hourly data set . . . 25
Figure 2.19 Cloud coverage histogram - hourly data . . . 25
Figure 3.1 Working principle of 10-fold cross validation . . . 30
Figure 3.2 Predefined split cross-validation . . . 30
Figure 3.3 Unseen systems split cross-validation . . . 31
Figure 3.4 Bias vs. variance trade-off . . . 32
Figure 3.5 Bias vs. variance - learning curves . . . 32
Figure 3.6 Accuracy and precision for regression . . . 33
Figure 4.1 XGBoost feature importance for hourly rough data . . . 42
Figure 4.2 Extreme Gradient Boosting (XGBoost) feature importance for hourly detailed data . . . 43
Figure 4.3 XGBoost learning curve at the beginning of the learning process - hourly rough data set . . . 44
Figure 4.4 XGBoost learning curve - hourly rough data set . . . 44
Figure 4.5 XGBoost learning curve . . . 44
Figure 5.1 Mean Absolute Percentage Error (MAPE) distribution calculated for each system individually . . . 48
Figure 5.2 RMSE distribution calculated for each system individually . . . 48
Figure 5.3 Comparison of percentage error distributions . . . 49
Figure 5.4 RMSE distribution calculated for each system individually . . . 50
Figure 5.5 MAPE distribution calculated for each system individually . . . 50
Figure 5.6 Comparison of different weather factors with respect to residual error . . . 52
Figure 5.7 Observed yield vs. absolute percentage error . . . 54
Figure 5.8 Error analysis with respect to time of the day . . . 54
Figure 5.9 Error analysis with respect to months . . . 55
Figure 5.10 Residuals vs. system size . . . 55
Figure 5.11 System size vs. MAPE . . . 56

Figure 5.12 System size vs. RMSE 56
Figure 5.13 Percentage error distributions before and after adding MAPE and weight-based normalization to the XGBoost training process 56
Figure B.1 ElasticNet feature importance for daily data 81
Figure B.2 Random Forest feature importance for daily data 82
Figure B.3 XGBoost feature importance for daily data 82
Figure B.5 Random Forest feature importance for hourly rough data set 82
Figure B.4 ElasticNet feature importance for hourly rough data set . . . 83
Figure B.6 ElasticNet feature importance for hourly detailed data set . . 83
Figure B.7 Random Forest feature importance for hourly detailed data set 83
Figure D.1 ElasticNet learning curve - daily data set 89
Figure D.2 ElasticNet learning curve - daily data set 90
Figure D.3 Polynomial Regression learning curve - daily data set 90
Figure D.4 Random Forest learning curve - daily data set 91
Figure D.5 Random Forest learning curve - daily data set 91
Figure D.6 XGBoost learning curve - daily data set 91
Figure D.7 XGBoost learning curve - daily data set 92
Figure D.8 ElasticNet learning curve - hourly rough data set 92
Figure D.9 Polynomial Regression learning curve - hourly rough data set 93
Figure D.10 Random Forest learning curve - hourly rough data set 93
Figure E.1 Support Vector Regressor (SVR) working principle [Bishop, 2006] 97
Figure E.2 Distribution of percentage error of Gradient Boosting with quantile loss function trained on data with daily resolution . 99
Figure E.3 Topology of a neural network 100
Figure E.4 Long Short-Term Memory (LSTM) cell and its operations [Phi, 2018] 100

LIST OF TABLES

Table 1.1 Overview of different temporal variability and forecast horizon scales and the associated purposes (target) [Elsinga and van Sark, 2017] . . . 5
Table 2.1 All available features . . . 17
Table 4.1 Results comparison for all models for daily data set . . . 40
Table 4.2 Results comparison for all models for the hourly rough data set . . . 40
Table 4.3 E-metrics for hourly rough data set . . . 41
Table 4.4 Results comparison for all models for the hourly detailed data set . . . 41
Table 4.5 E-metrics for the hourly detailed data set . . . 41
Table 4.6 Comparison of XGBoost metrics for daily aggregated results . . . 42
Table 4.7 Top 10 systems with the worst predictions by Solar Monkey . . . 45
Table 4.8 Top 10 systems with the worst predictions by the best XGBoost model . . . 45
Table 5.1 Individual system metrics comparison for Solar Monkey and XGBoost trained on randomly split data . . . 47
Table 5.2 Comparison of individual system metrics for Solar Monkey and daily aggregated XGBoost model for previously unseen systems and identical time steps in training and testing data . . . 49
Table 5.3 Comparison of individual system metrics for the Solar Monkey model and XGBoost trained using *GroupShuffleSplit* and evaluated on future time steps and **seen** systems . . . 51
Table 5.4 Comparison of individual system metrics for the Solar Monkey model and XGBoost trained using *GroupShuffleSplit* and evaluated on future time steps and **unseen** systems . . . 51
Table 5.5 Comparison of performance for XGBoost and persistence model - hourly XGBoost model with unseen systems in the test set . . . 53
Table 5.6 Comparison of performance of XGBoost and Solar Monkey model - configuration with unseen systems in the test set . . . 53
Table 6.1 The best model set of features with their corresponding weights 60
Table 6.2 Performance comparison of one XGBoost model for multiple systems and XGBoost model built for each system individually [Visser, 2018] . . . 61
Table C.1 Optimal configuration of ElasticNet hyperparameters - daily data . . . 85
Table C.2 Optimal configuration of Random Forest hyperparameters - daily data . . . 85
Table C.3 Optimal configuration of XGBoost hyperparameters - daily data 85
Table C.4 Optimal configuration of ElasticNet hyperparameters - hourly rough data . . . 86
Table C.5 Optimal configuration of Random Forest hyperparameters - hourly rough data . . . 86
Table C.6 Optimal configuration of XGBoost hyperparameters - hourly rough data . . . 86
Table C.7 Optimal configuration of ElasticNet hyperparameters - hourly detailed data . . . 86
Table C.8 Optimal configuration of Random Forest hyperparameters - hourly detailed data . . . 86

Table C.9 Optimal configuration of XGBoost hyperparameters - hourly detailed data 87
Table F.1 Used hardware configuration 103
Table F.2 Versions of the utilized libraries 103

ACRONYMS

AI Artificial Intelligence . . . 5
APE Absolute Percentage Error . . . 51
ARIMA Autoregressive Integrated Moving Average . . . 6
ARMA Autoregressive Moving Average . . . 5
CNN Convolutional Neural Networks . . . 6
EDA Exploratory Data Analysis . . . 21
GHI Global Horizontal Irradiance . . . xi
GPOA Global Plane of Array . . . xi
HMM Hidden Markov Model . . . 9
LIME Local Interpretable Model-agnostic Explanations . . . 7
LSTM Long Short-Term Memory . . . xii
MAE Mean Absolute Error . . . v
MAPE Mean Absolute Percentage Error . . . xi
MBE Mean Bias Error . . . 34
ML Machine Learning . . . 3
MLP Multilayer Perceptron . . . 6
MPP Maximum Power Point . . . 7
MSE Mean Squared Error . . . 34
NARX Nonlinear Autoregressive Exogenous . . . 6
NEI Not Enough Information . . . 77
NLP Natural Language Processing . . . 101
NN Neural Network(s) . . . 6
NOCT Nominal Operating Cell Temperature . . . 60
NRMSE Normalized Root Mean Squared Error . . . 35
PCA Principal Component Analysis . . . 8
PV Photovoltaic . . . 3
RFE Recursive Feature Elimination . . . 27
RMSE Root Mean Squared Error . . . v
SS Skill Score . . . 9
STC Standard Test Conditions . . . 25
SVR Support Vector Regressor . . . xii
XGBoost Extreme Gradient Boosting . . . xi

1 INTRODUCTION

Since the industrial revolution in the XVIII-th century global population, knowledge and economy have not grown linearly anymore, but have changed exponentially. The expected human lifetime in the Netherlands has changed from 48.5 years in 1900 to 81.8 years in 2019 according to [Gapminder Foundation, 2020]. In the same period the percentage of global population living in extreme poverty shrinked from 72% to 9% [Rosling et al., 2018]. Availability of cheap and abundant energy brought unprecedented era of prosperity and welfare. Even though they were not available to all, vast majority benefited from their presence. Cheap energy largely contributed to the interconnected World we know today, with fast and affordable transportation and powerful computers that are able to both boost scientific developments and provide entertainment.

However, spectacular developments came at high cost. Emissions caused by transportation and industry owe increasing the average air temperature, leading to melting of the ice caps and rise of the sea level. Due to climate change multiple habitats were irreversibly destroyed. The amount of litter produced worldwide is so large that even if humanity disappears from Earth, signs of its presence will be still visible in soil. According to [The Economist, 2014] another mass extinction has already started and a new geological era, the *antropocen*, has already began. Despite the efforts of European Union, most of the world is still focusing on economic growth and sustaining its basic needs, unable or unwilling to tackle the climate change on global scale. Paris agreement ratified in 2015 and signed by 176 countries aims to keep the global temperature rise below 2°C and shows that politicians have increasing awareness of the issue. However, soon after signing it one of the world's largest economies, the United States, has withdrew from the pact. It seems that media attention and political actions are disproportionate to the taken measures.

Luckily, new and promising technologies might come for rescue. Between 2010 and 2019 the market of photovoltaic modules raised by 32 % annually [Fraunhofer Institute for Solar Energy Systems, 2020] being the largest hope to tackle climate change. With monetary impact being the crucial factor shaping human actions and policies, solar energy can have a profound impact worldwide. Solar power is abundant, affordable, easily scalable and does not emit CO_2 during its operation. However, mass utilization of solar modules has a major challenge of intermittency of supply, which makes it difficult to maintain power balance, to plan reserve capacity and complicates market bidding. Therefore, Photovoltaic (PV) module yield forecasting is an important factor facilitating the energy transition and supporting investment in solar energy. Accurate forecasts decrease energy yield uncertainty, therefore reducing generation-load mismatch in the power grid and decreasing investment risk. Yield nowcasting (monitoring) ensures early anomaly detection preventing economic losses and contributing to financial security of PV system owners. Until recently the described tasks were difficult due to lack of suitable models. Analytical equations hold in laboratory environment, but often fail to predict yield in the field, with insufficient information or with large data resolution. Taking continuous measurements of all required parameters in situ is not a common practice due to high associated costs. With insufficient data it is the emergence of Machine Learning (ML) and deep learning techniques which allowed the creation of more accurate and precise models.

1.1 LITERATURE STUDY

This section provides brief description of possible approaches to PV yield nowcasting and forecasting along with references to corresponding literature. Although the focus of this Master thesis is photovoltaic yield *nowcasting*, many scientific publications described in this chapter refer to yield *forecasting*. The reason is the similarity between these two problems. However, some differences exist and are stated explicitly whenever that is the case. The task of yield prediction depends on the provided inputs. In this thesis nowcasting is defined as predicting the present or very near past. It helps to determine what is the solar yield a PV system should produce at the given time and weather conditions. Nowcasting is used for early anomaly detection. If expected weather parameters are provided instead of historical values, the described models will predict future yield (forecast). An overview of analyzed papers in these fields was presented in appendix A. The presented review refers to research performed in multiple countries and in variety of climates, but due to great abundance of related work, by no means is exhaustive. Scientific publications regarding the topic of interest can be divided into the following categories: studies comparing several forecasting techniques and striving to pick the best one of them, studies focusing on PV systems anomaly detection, studies providing overview of related subjects such as feature engineering and finally comparative studies written to keep track of the progress in the field. Several other types of articles such as machine learning theory and irradiance or cloud coverage forecasting are also relevant for solar yield monitoring. Machine learning and solar yield forecasting are subject to quick changes. Therefore, this report usually refers to the most recent publications in these fields.

Comparative studies and their influence on this work The single most important comparative study is the one by [Yang, 2019] who presented analysis of 79 scientific papers related to solar yield forecasting. The publication contains valid criticism of current solar forecasting research. The author claims it is difficult to compare results due to lack of standard benchmark, lack of open-source access to the utilized data, evaluation on small data sets, intentional hiding of the shortcomings, and overwhelming abundance of the related literature. The article provides a set of reporting rules called *ROPES* which is proposed "to assist researchers in preliminary assessment of a publication, and to set guidelines for future research" [Yang, 2019]. **R** stands for *reproducible* and reminds that the used data base should be freely published, so other researchers can pursue a different approach and be able to directly compare the results. **O** stands for *operational* and implies that motivation of research should be clear, so the grid operator could easily understand its relevance for grid management and generators dispatch. **P** stands for *probabilistic* and implies that probabilistic methods are more desirable and should be promoted. **E** is herein used to denote *ensemble* NWP (physical ensemble), ensemble learning (machine learning ensemble) or forecast combination (statistical ensemble). Finally, **S** stresses the importance of *skill score* which shows relative improvement with respect to persistence method. It underlines also the importance of common benchmark for all results. Previously to [Yang, 2019], International Energy Agency issued an article structuring the research on solar yield forecasting [Pelland et al., 2013]. Also [Antonanzas et al., 2016] and [Sobri et al., 2018] compared multiple articles related to solar yield forecasting including comparison of the used data time resolutions, input variables, locations, and methods.

Described guidelines were adjusted to the case study of PV yield monitoring and implemented to provide transparency and maximize the scientific value of this study. Unfortunately, the work presented in this thesis cannot be freely reproduced, as performed study laid foundations for a launch of a commercial product. Therefore, easy reproducibility is not aligned with the best interest of Solar Monkey. Nev-

ertheless, a detailed description of inputs, their resolution and units are provided. Process of feature engineering, the exact used libraries, hardware specifications and model computational times are also described. Feature importance for different models is presented. Described work is focused on yield monitoring, rather than forecasting, hence it is of no use for the power system operator. However, to provide an overview, different forecast horizons together with their applications are presented in table 1.1. Unfortunately, none of the methods described in this work

Table 1.1: Overview of different temporal variability and forecast horizon scales and the associated purposes (target) [Elsinga and van Sark, 2017]

Time horizon	<15 min	15 min-h	h-day	>day
Target				
Power balance/quality	x			
Reserve capacity planning	x	x		
Load following	x	x		
Market bidding		x	x	x
Base-load planning			x	x

calculates probabilities. That is desirable for facilitating anomaly detection, but was not investigated due to time limitations of the project. This research is ensemble-oriented, as it focuses on XGBoost algorithm taking advantage of gradient boosting (appendix E) and random forest (appendix E). Skill score values with respect to **simple persistence** method are provided for easy comparison with other nowcasting and forecasting research. According to [Yang, 2019]: "based on what has been published by the field leaders, it should be noted that a forecast skill of 0.3–0.5 is the usual improvement that one should expect. Occasionally, due to an extremely effective predictor, e.g. data from a station in the upwind direction to the focal station, higher skills such as 0.5–0.7 can be achieved. Any outrageous forecast skill reported, such as >0.7 for the entire year, is most likely due to some computational or algorithm-design error". Mentioned values refer to forecasting which uses uncertain inputs and therefore has lower skill scores than monitoring. [Sanfilippo et al., 2016] reported 44.92% skill score improvement over the persistence model, for solar nowcasting.

Possible approaches towards forecasting Studies such as [Antonanzas et al., 2016], [Sobri et al., 2018] and [Raza et al., 2016] provide an overview of all the methods applicable to solar yield monitoring. Three main approaches emerge from these works: analytical (called also physical), statistical, and Artificial Intelligence (AI) approach. The latter splits into deep learning and ensemble learning. Analytical models were explored by the author of this thesis during a summer internship. Most of the analytical models described in [Smets et al., 2016] were implemented together with irradiance decomposition model BRL [Ridley et al., 2010], inverter efficiency model SNL [King et al., 2007] and complex shading, but failed to provide accurate daily predictions using hourly data resolution. Utilization of per minute or per second data is not feasible, as most of the analysed systems are small-scale residential installations which lack complex measuring devices. Additionally, processing e.g. per second data would require around 3600 more computational power than in case of hourly data. This would either significantly increase computational time or drastically increase hardware requirements. Physical (analytical) models require information about electrical configurations, and therefore are prone to errors. PV systems often consist of more than one type of modules which is difficult to track in large data sets. As a result of these findings, the analytical approach towards PV monitoring was abandoned. Nevertheless, the analytical models should not be treated as incorrect. They have proven their value for high granularity data and are the method of choice when there is little uncertainty regarding inputs.

Second possible approach is utilization of statistical methods such as Autoregressive

Moving Average (ARMA), Autoregressive Integrated Moving Average (ARIMA), and Nonlinear Autoregressive Exogenous (NARX). They focus on measuring correlations between dependent and independent parameters and can be split into stationary and non-stationary or linear and non-linear. Stationary time-series fluctuates around its mean [Sobri et al., 2018]. According to [Sharadga et al., 2020] this approach provides worse results than the AI approach. ARIMA is not considered to be a method of choice in unstable climate, as proven by [Isaksson and Conde, 2018]. [Raza et al., 2016] value these methods more, as they claim that "ARMA and/or ARIMA can be used where fewer meteorological parameters are available as model input for accurate forecasting PV output power". According to [Massaoudi et al., 2019] ARMA and NARX have proven their utility for short term forecasting and are known for their simplicity. They do not require large database for training and have low hardware requirements. Due to varying opinions about the suitability of these methods, ARMA, ARIMA and NARX were not developed in this project. Additional reason is that time series methods require different data pre-processing than most of the ML methods. Working principles of ARMA and ARIMA can be found in [Raza et al., 2016] and [Sobri et al., 2018].

Finally, there is AI approach which is the subject of this research. According to [Antonanzas et al., 2016] it "proved superior when compared to the parametric approach. Most recent papers used machine learning techniques, due to the ease of modeling without the need of knowing PV plant characteristics". All the described models except for benchmark persistence model and current Solar Monkey's model, follow this path. The best performing ML methods split into deep learning which is based on neural networks and ensemble learning which combines many weak learners (predictors) into one strong learner. According to [Maitanova et al., 2020] 24% of researchers uses Neural Network(s) (NN) for predicting PV power. Dominant methods are Convolutional Neural Networks (CNN), Multilayer Perceptron (MLP) and LSTM neural networks. Especially the latter was reported to have supreme performance over other algorithms according to [Abdel-Nasser and Karar, 2017] and [Gensler et al., 2016] who suggested combining LSTM and CNN neural networks for further improvement in performance. Attempts to build LSTM neural network by the author of this thesis were described in appendix E. According to [Sobri et al., 2018] "in comparison with other approaches i.e., one-diode, analytical, polynomial regression and multiple linear regression methods, artificial neural networks demonstrate the lowest mean relative error". They concluded that "the metrics assessment shows that AI models could decrease the error compared to other statistical approaches". Usage of neural networks is also recommended by [Tai, 2019] over linear regression, decision trees, Gaussian process and computation of point resemblance. Based on the described findings deep learning and machine learning were chosen as methods of choice. ML models were developed first due to lower complexity and shorter development time. Simple models of LSTM neural networks were built, but were abandoned due to time constraints of this project.

Next to the introduced three main approaches, hybrid methods exist, e.g. [Durrani et al., 2018] combined data pre-processing using sophisticated analytical models with artificial neural networks reporting MAPE of 3.4 % for sunny days and 23 % for cloud days. Similar approach was pursued by [Ogliari et al., 2018] who used PV module equivalent circuit and artificial neural networks for yield forecasting and reported skill score equal 47%. An overview of available methods prepared by [Raza et al., 2016] can be seen in figure 1.1.

Gradient Boosting for Solar Yield Nowcasting Among many researchers dealing with gradient boosting are [Nikolaou et al., 2017], [Massaoudi et al., 2019] and [Visser, 2018]. [Nikolaou et al., 2017] also utilized shading calculations, as he assumed that each string consists of 12 modules and passed a fraction of shaded

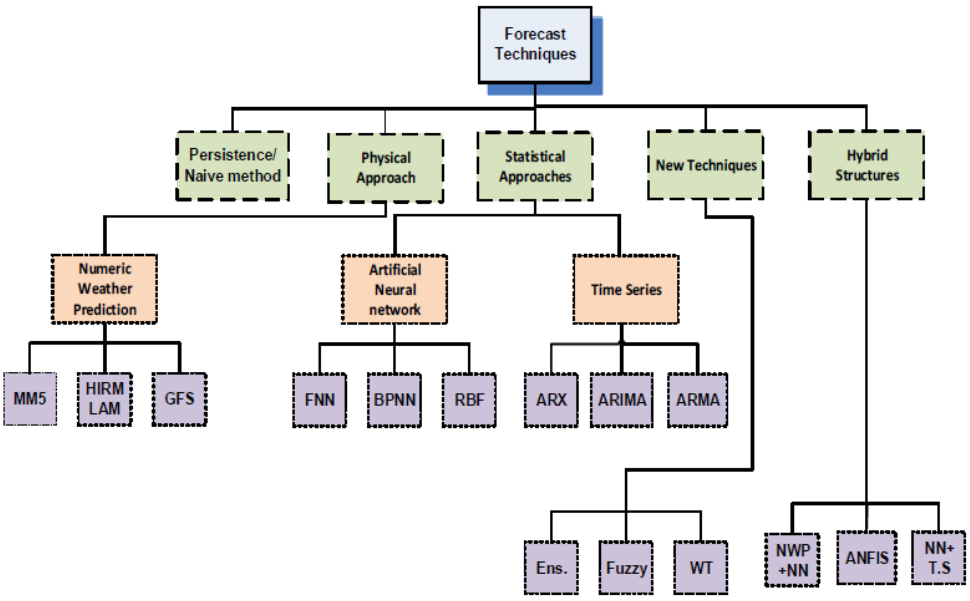


Figure 1.1: Classification of forecasting techniques [Raza et al., 2016]

modules as input to XGBoost model. E.g. if 2 out of 12 modules are shaded, shading factor equals 2/12 = 0.17 pu and is provided as an input. Interestingly, he found out that large data set is not required for XGBoost to obtain good performance. It is important to mention that [Nikolaou et al., 2017] utilized equivalent circuit and focused on Maximum Power Point (MPP) tracking under partial shading conditions. [Massaoudi et al., 2019] built blended models combining Local Interpretable Model-agnostic Explanations (LIME), ElasticNet and XGBoost to obtain feature importance.

Lack of Large Database Analysis Attempts to structure the research on solar yield forecasting are very relevant, as despite overwhelming abundance of literature little progress is being done. For example, only [Elsinga and van Sark, 2017] provided analysis for 202 rooftop PV systems while no other studies utilized data for more than 21 systems. Issue of small data sets was noticed also by [Theocharides et al., 2018] who wrote that "most approaches are not fully tested and verified on large amount of field data and different technologies, there is yet no complete universal forecasting model and methodology to ensure accurate forecasts according to the technology and location". Most of the researchers motivate their study by contributing to improved generator dispatch, power quality effects mitigation, and reducing secondary reserve capacity [Theocharides et al., 2018], but if their results are not validated for multiple systems, they are not reliable enough for upscaling. Large scale data acquisition is costly and might be difficult for universities and research institutions. Lack of coordination between researchers and the industry might result in unnecessarily long search for optimal solutions and little implementation. This study aims to fill this literature gap by providing analysis of one year data for 1102 PV systems.

Choice of inputs Next to investigating comparative studies described before, an independent literature study was performed. It focused on determining which algorithms are used the most often for PV yield forecasting, what kind of inputs are utilized, what is their time resolution and which metrics are used for evaluation. Many researchers utilized values of voltage and current as inputs to their models [Sharma et al., 2020], but these information were not available during this project. Other researchers used past solar radiation as a feature [Massaoudi et al., 2019], [Maitanova et al., 2020]. There is variety of approaches and choosing the right one depends on the forecast horizon and type of available data. According

to [Maitanova et al., 2020] it is possible to generate predictions without PV system information, basing mostly on weather data and values of the historical generated power. Utilized inputs had 5 minute resolution and consisted of historical weather measurements and maximal measured PV power in the previous five days, considering only the time for which the predictions are made. It is worth noticing that this approach does not require information about solar radiation. Further comparison of used algorithms and input features can be found in appendix A. Many researchers included air humidity and pressure which are less relevant from solar engineering perspective. The most typical features are: GHI, GPOA, module temperature, wind speed and cloud coverage. Less typical features are maximal yield in the preceding five days, PV power from the same time instant a year before, electrical characteristics such as MPP voltage and current. In the analyzed literature no study utilizes obstacle views nor calculates albedo.

Cloud coverage information According to [Anagnostos et al., 2019] "predicting PV output under clear sky conditions is a trivial task". Rapidly changing cloud coverage has significant influence on PV modules output and is the biggest challenge of solar yield forecasting. An extensive overview of physical methods for cloud coverage analysis was provided by [Sobri et al., 2018] who described usage of sky imagery and satellite imaging. They distinguished four elements of sky imagery analysis: obtaining sky images, analyzing them to recognize clouds, estimating cloud motion vector using consecutive images and utilizing obtained location and motion vector data for making irradiance, cloud coverage and power predictions. Other researchers use different techniques and inputs to incorporate cloud coverage information in their models. An interesting approach was proposed by [Gandoman et al., 2016] who calculated seasonal variation in oktas independent of geographical location and used them to calculate probability of power variations. The model was trained on 20 years of cloud coverage data which increased reliability of this study. [Zhang et al., 2018] also sees the importance of cloud analysis from sky images, as he states that "constantly changing clouds are still quite hard to model and create inaccurate future power output predictions". Lack of sky images and high resolution cloud coverage information was a major concern in this project.

Anomaly Detection Nowcasting tool developed in this project will be used by Solar Monkey for anomaly detection. Therefore, a description of monitoring related literature is presented. A review of solar plant monitoring techniques was created by [Ejgar and Momin, 2017] who described how yield data from different strings are gathered and compared to calculate correlations and detect anomalies. Other described approaches include Markovian models, Gecko clustering algorithm or anomaly detection libraries such as *EGADS* in Java or *luminol* in Python [Ejgar et al., 2016]. Several researchers described possible approaches towards anomaly detection. [Aziz et al., 2020] investigated CNN for this task and categorised faults into line-to-line, line-to-ground, open-circuit, hot spots, and environmental effects. He concluded that fault classification significantly increases when MPP tracking data is used. A recent study by [Taghezouit et al., 2020] describes a superior method of anomaly detection in PV systems using Principal Component Analysis (PCA) and Kernel Density Estimation. Unfortunately, requirement for obtaining electrical data limited the application of this study. Different path towards anomaly detection was pursued by [Branco et al., 2020] who utilized only yield values and identified five kinds of faults: daytime zero production, low maximum production, daytime shading, sunrise/sunset shading and sub-optimal orientation. His method utilizing yield time series data and was not chosen due to low reported detection rate and large number of false positives. The latter means that correct values were treated as anomalies and removed leading to loss of high quality data and causing discontinuity. Finally [Rivai et al., 2020] suggested usage of low-cost sensors for anomaly detection. This path is currently explored by Solar Monkey.

One might wonder why this study focuses on regression if its main objective is anomaly detection. There are several reasons for that. Firstly, using regression makes it is easy to measure how large model errors are. Secondly, regression models can be used for yield forecasting if predicted weather data is provided. This means the same model can be used for two different tasks. In this work a simple approach of comparing predictions with observed values was pursued. A more interesting approach would be using probabilistic supervised learning. Differences between it and regular regression are described by [Schuler, 2019]. Classification algorithms could also be utilized for this task.

Solar Monkey's internal research The most relevant studies to the research described in this report were performed by interns at Solar Monkey, Eefje Visser and Yves van Montfort who focused on yield forecasting and data cleaning, respectively. Significant overlap exists between this work and the work by [Visser, 2018] who developed the similar models as in this project and additionally investigated SVR and MLP. Similar features including past yields, shading calculations, and weather parameters were used. The approach described by [Visser, 2018] focuses on creating separate models for each system and allows to learn their individual parameters such as ageing and changing obstacle views (e.g. growing trees). E. Visser assumes usage of general model when no data for a particular system is available and replacing it with a system-specific model after 6 months. That is fundamentally different than this project, as it is assumed that general model is able to learn cross-system parameters such as inverter efficiency, module decay per year, system latitude and longitude, module type and efficiency. Usage of inverter efficiency was recommended by [Visser, 2018] and implemented in this project. She argued also that PV systems generate large amounts of data throughout their lifetimes and suggested usage of incremental learning: "we wish to update the model with incoming data, without having to construct the model from the ground up each time it is updated". That is arguable, as this way models would not be able to see yearly patterns in the data. E. Visser acknowledged also that "due to its very short training time, performing updates by calculating the complete model from the ground up is a feasible option for the XGBoost algorithm". Moreover, large data sets tend to favor neural networks which are known to maintain steep learning curves longer than machine learning models. Large data set is one of the most important assets of Solar Monkey and distinguishes its models from other developments. These fundamental differences in approach resulted in many decisions taken in this work. It is believed that training separated models for each system is cumbersome, as of April 24th 2020, total number of systems monitored by Solar Monkey has exceeded 10,000. In this study 1,102 systems were investigated, contrary to 81 systems analyzed by Eefje. Another large difference is the calculation of model metrics which is more complex in the case of one multi-system model. In order to provide a cross research comparison, next to Skill Score (SS) E-metrics developed by [Visser, 2018] were calculated. An obstacle of direct comparison is that models described in that work utilized 15-min data resolution which gives them significant advantage.

Other Solar Monkey intern, Yves van Montfort performed very relevant study regarding data cleaning. He used the same data set as utilized in this project and discovered five different data quality issues: missing data, low outliers, high outliers, constant and lagging yield. Hidden Markov Model (HMM) was utilized to detect all mentioned kinds of corrupt data and reached 99.6% accuracy and 95.4% precision for this task [Solar Monkey, 2020]. These results are the very reason for applying the developed HMM in this study. Further information on data quality issues can be found in section 2.1.

1.2 RESEARCH OBJECTIVES

The aim of this report is to present a method to create a very accurate and precise PV yield nowcasting model. It is intended to confirm that AI is a suitable tool for this task and outperforms the currently used analytical model. It was also investigated whether the presented methods are cost efficient and can fulfill time and cost requirements of the industry. Detailed research questions can be found below:

- *RQ1: Which machine learning algorithm provides the best performance for the task of PV yield monitoring?*
- *RQ2: What are the common variables chosen by all models and what is the best set of inputs?*
- *RQ3: Should machine learning methods replace the analytical approach?*
- *RQ4: Is developing a single general model better than developing models for all systems individually?*
- *RQ5: Is GPOA precalculation beneficial for ML models? Would using its raw components provide better results?*

The goal of this thesis is to utilize and verify the existing knowledge and provide description of cutting-edge commercial product development, as well as to identify core physical parameters influencing photovoltaic yield.

1.3 SOLAR MONKEY

The research was founded by Solar Monkey company backed up with PVMD group. Its mission is to provide financial security to the owners of PV systems and to decrease carbon dioxide emissions through preventing PV systems' downtime. Solar Monkey offers software allowing system design and providing year-ahead yield predictions. It sells also monitoring service for the existing installations and yield guarantees (a form of insurance). The company grows rapidly and so far operates in the Netherlands, Belgium and Spain. In 2019 Solar Monkey left startup incubator *YES! Delft* and moved to *The Hague Tech* where it currently employs more than 20 people. It was also recently nominated for *Rising Star* category of Deloitte Technology Fast 50 [Solar Magazine, 2019].

1.4 OUTLINE

This document will be presented in the following structure. After presenting the related work in chapter 1, description of model inputs and feature engineering is provided in chapter 2. Theoretical background related to machine learning theory together with discussion about metrics can be found in chapter 3. Comparison of results for all three analyzed data sets is presented in chapter 4. More detailed analysis of the best model selected based on results comparison can be found in chapter 5. Chapter 6 contains project conclusions. Finally, chapter 7 gives recommendations for further product development in Solar Monkey and for further academic research.

Information complementary to this study can be found in appendices. Appendix A contains a review of selected scientific articles including utilized models, metrics and novel aspects. Appendix B presents feature importance graphs, appendix C contains hyperaparameter configurations for each model and data set. Appendix D

provides learning curves corresponding to all analyzed algorithms and data sets together with their descriptions. Appendix E provides insight into selected ML models working principles and appendix F provides information about hardware utilized in this project together with versions of the utilized libraries. Finally, appendix G contains the first draft of a scientific publication based on this project.

2 DATA

In this chapter a description of inputs to machine learning models is provided. This project utilizes two main data resolutions, daily and hourly, and three separate data sets. It is intuitive that finer data resolution can provide higher quality results. However, hourly data are available within Solar Monkey only for systems with *Solar Edge* inverters. Remaining systems utilizing *Goodwee*, *Solcast* and other inverter brands can log only daily system yield. First of the three mentioned data sets has daily resolution, second contains hourly data with features used for GPOA calculation and is often referred to as *hourly rough* data set. That is different than the last, *hourly detailed*, data set which instead of raw components contains pre-calculated GPOA values. Each data set has a different purpose. First, daily data set, was used to prove that AI is a better approach than current Solar Monkey's method. Second, hourly rough, was used to show the increase in performance due to improved data resolution. Finally, hourly detailed, was used to verify whether solar engineering theory used for data pre-processing can boost performance of ML models. The main objective of the project was to create as good monitoring service as possible and it was assumed that it can be obtained using data containing GPOA values. In order to provide valid comparison, all three data sets contain data for the same PV systems.

Yield forecasting is a *supervised* machine learning problem which means it utilizes a set of correct answers, called *labels*, for model training. This is contrary to *unsupervised learning* which focuses on finding unknown patterns in the data. Because this project utilizes supervised learning, two matrices are required for training purposes: feature matrix X containing weather and system parameters together with historical yields and target matrix Y containing corresponding yields. Description of their pre-processing is the main objective of this chapter.

2.1 DATA QUALITY

In order to ensure that model performance is maximized only clean data should be used for training and testing which is a major challenge. Data used in this project comes from three different sources. Weather data were obtained from KNMI, system data were provided by the installers and yield data were sent by the inverters. Data from each source suffer from different quality issues. The most important errors are related to system data such as number of modules, their orientation and tilt, type of inverter, string configurations and other information provided by the system installers. It is not verified and customers put little attention to filling in those values. Therefore, several systems have wrong string configurations. That is a very serious data quality issue, as no algorithm can perform well using incorrect inputs. Detection of these kind of errors before creating a working model is difficult, if not impossible. Next, KNMI data often contains missing values for cloud coverage which is an important parameter influencing solar yield. Apart from that, no other data quality issues in those inputs were detected. Finally, yield data are gathered by measurement devices which are not reliable for 100% of time, therefore, it might happen that measurements are skewed or missing. PV modules can be shaded or damaged which significantly decreases their yield, despite favourable environmental conditions. The same applies to PV systems down for maintenance.

Even if yield during fault or partial shading is correctly measured, it should not be used for ML models training. Training set should consist of samples corresponding to uninterrupted operation, so predictions corresponding to malfunctions have large error. In other words, a perfect nowcasting model **should** be mistaken in case of partial shading or system malfunction. Training examples containing corrupt data decrease overall performance of machine learning algorithms. Insightful research on this issue was performed by [Solar Monkey, 2020] who classified five types of corrupt data: low outliers, high outliers, missing data, constant yield and time shifted data. These issues are tackled using two methods depending on data resolution.

2.1.1 Daily Inputs

Low and high outliers occur when the resulted and expected yields significantly differ. In this case current Solar Monkey's predictions are used as reference. Missing data is caused by poor inverter wi-fi connection and yield values occur as zero during non-zero GHI above 10 W/m^2. Based on applying SNL model [King et al., 2007] on several PV inverters, that threshold is considered to allow PV modules to produce power exceeding inverter startup power. Constant yield corruption occurs when yield values are equal and non-zero for several consecutive time steps. Likelihood of such event in reality is extremely low, as weather keeps changing and yield values are measured with mWh precision. Examples of corrupt data can be seen in figure 2.1. HMM was able to detect and label corrupt information with estimated

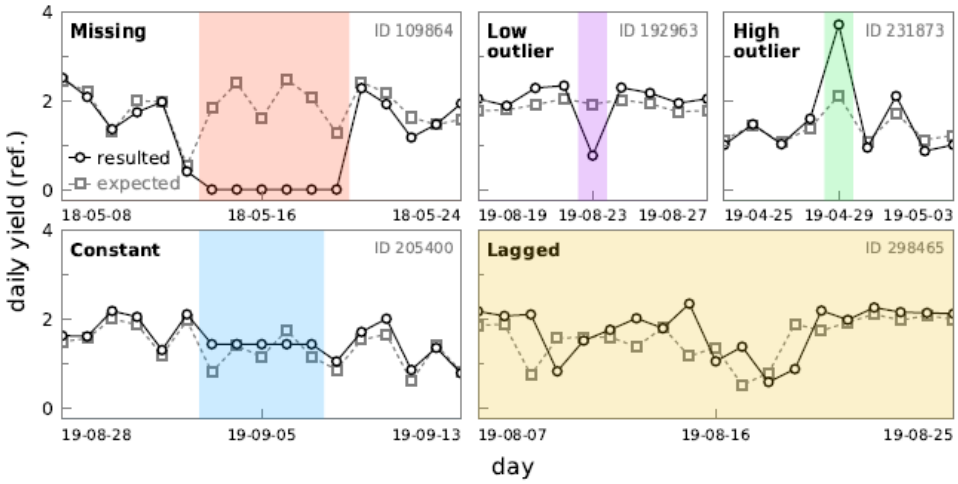


Figure 2.1: Types of data corruption [Solar Monkey, 2020]

sensitivity of 99.9%. This algorithm returned probabilities of data being clean. Only labels with likelihood above 99% were kept in the data set. Even though this value is very conservative, only 1.11% of all yield values (labels) had lower probabilities and were removed. Due to time constraints no sensitivity analysis was conducted, but due to HMM usage XGBoost's RMSE dropped from around 1.45 to around 1.25 *kWh/system/day*. Low and high outliers were not removed, because HMM can detect them with low precision equal to 34.9%. That might be caused by relying on current Solar Monkey's model for outlier detection. Its low performance seems to harm HMM's precision. Data cleaning tool was developed and validated by [Solar Monkey, 2020] and it was merely plugged-in to ML models by the author of this thesis.

2.1.2 Hourly Inputs

In the hourly data set a simpler approach was pursued. Labels were filtered assuming that if three consecutive non-zero labels are equal, they are corrupt (constant yield). If there are three consecutive zero yields, they all are assumed to be missing. If non-zero yield is exactly the same as 24 hours ago it is lagging. This assumption is safe only because yields are given with accuracy up to mWh. Therefore, likelihood of two identical yields on consecutive days is very low. To identify high outliers, yield values higher than three standard deviations of all yield values for a particular system were detected [Ilyas and Chu, 2019]. From those only morning and evening cold months outliers (before 11 AM or after 5 PM, in months October - March) were included. No method of low outliers identification was developed. The results of data filtering can be seen in figure 2.2. According to the described filters 4.16% of all hourly yield data is corrupt. Unfortunately, after data cleaning performed using the

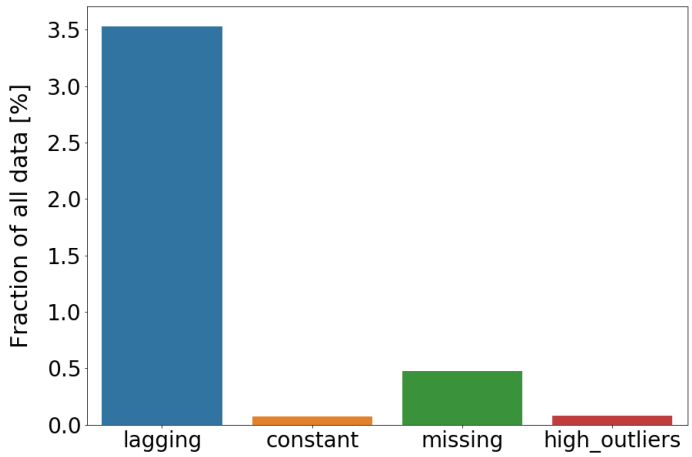


Figure 2.2: Fraction of corrupt data

described filters, models tend to perform worse. Therefore, corrupt yield examples were counted, but were not removed. Without sensitivity analysis it is impossible to determine whether these assumptions actually hold. The following graphs were presented for overview and are intended to provide insight into hourly data quality issues which should be tackled in the future. Implementation of the HMM to hourly data was not possible due to project time constraints and because the model was not available at the beginning of the project. Moreover, Solar Monkey has recently released a device[1] which logs yield data not suffering from the described issues (J. Donker, personal communication, May 20th, 2020). Therefore, hourly data cleaning was not investigated in detail.

Figure 2.3 depicts the issue of constant yield. It might seem that its value is equal to zero, but that is not the case. Until 2019-12-21 yield oscillates between 0 Wh and 15 Wh. It is possible to see that the same system was able to produce much more energy e.g. on 2018-12-25 when illuminated by similar irradiance as on 2018-12-19. Taking into account long period of close-to-zero yield it is reasonable to assume that this yield data is corrupt. The issue of missing yield was presented in figure 2.4 where yield is zero for nearly all day, despite favourable irradiance conditions. GHI is above 10 W/m^2, hence zero yield cannot be justified by not exceeding inverter self-consumption. Next, the issue of lagging yield was discovered. However, it was caused my the fact that KNMI data used UTC time while yield data logged by inverters used Dutch local time. After fixing this issue, despite careful manual investigation, no examples of lagging yield were found. Finally, high outliers were identified and depicted in figure 2.5. Despite significantly higher irradiance on Jan-

1 The device is produced by Xemex and utilizes GPRS communication

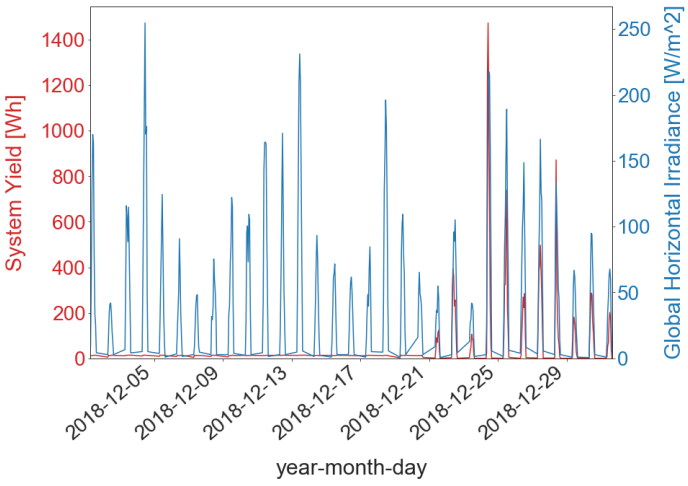


Figure 2.3: Example of constant yield

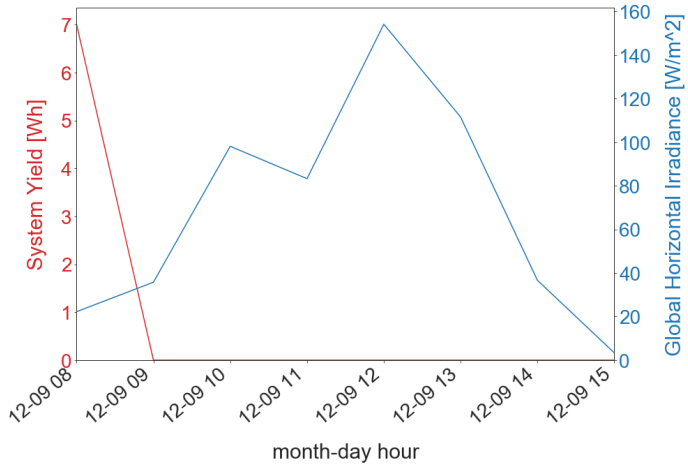


Figure 2.4: Example of missing yield

uary 21st, than on January 20th, the analyzed system harvests similar energy yield. This is suspicious, but insufficient to fully remove these samples from the data set.

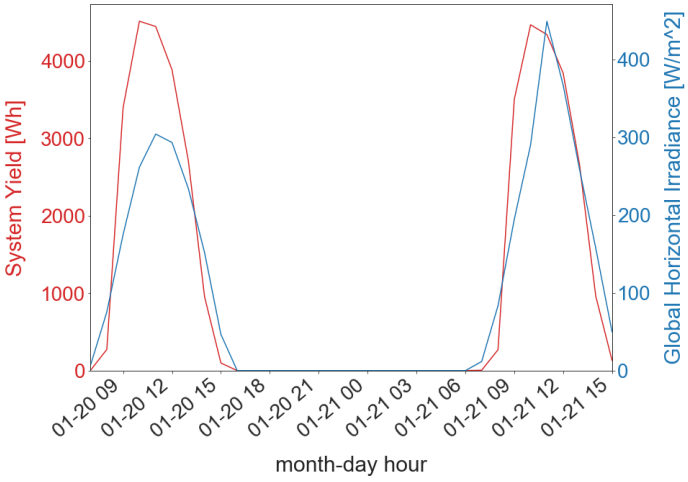


Figure 2.5: Example of high yield outliers

Described data filter makes detailed data quality analysis of all individual systems possible. This concept can be used for detecting data quality issues and is elaborated on in chapter 7.

2.2 FEATURE ENGINEERING

All variables influencing the outcomes of ML models are called *features* and the process of their pre-calculation performed after data cleaning is called *feature engineering*. In this section all operations performed both on daily and hourly inputs are described. List of all available features promising for the task of solar yield prediction together with their corresponding units can be found in table 2.1. However, not all of them were utilized in the final model, as some turned out to be correlated and therefore were removed. Correlations between features in table 2.1 are investigated in figures 2.6 and 2.7. The remaining features are uncorrelated which is desirable for machine learning models. According to [Alzahrani et al., 2017] night hours should be removed, as they do not have positive contribution to algorithms' performance. This was implemented, as it allows also to save memory and reduce computational time.

Table 2.1: All available features

Weather Features / Unit	Sun zenith	°
	Sun azmiuth	°
	Sun altitude	°
	precipitation	mm/h[1]
	ambient temperature	°C
	ground temperature	°C
	wind speed	m/s
	Global Horizontal Irradiance (GHI)	W/m^2
	Global Plane of Array (GPOA)[2]	W/m^2
	obstacle factor[3]	%
	cloud coverage	okta
System Features / Unit	Nominal Operating Cell Temperature	°C
	system inclination	°
	system orientation	°
	system latitude	°
	system longitude	°
	total watt peak	W
	module decay per year	-
	inverter efficiency	-
	parallel	T/F
	system age	days
	type mono	T/F
	type poly	T/F
	type thin film	T/F
Other Features / Unit	day of year	-
	yield 1 day before	kWh
	yield 2 days before	kWh
	yield 3 days before	kWh
	yield 4 days before	kWh
	yield 5 days before	kWh

1 or mm/day, depending on data resolution
2 only in hourly detailed data set
3 only in hourly detailed data set

Models accept only numerical features, hence categorical features such as *module type* had to be transformed by *getting dummies*. This function converts one column containing string values (in this case mono, poly or thin-film) into three columns, one for each unique string. Each of the newly created columns contains zeros and ones. For each row in the X matrix only one of the three columns contains one and the remaining two columns contain zeros. This includes hidden assumption that all modules in a system are of the same type. System age in days was calculated using system start date and time indices. Time series features of yields from previous five days were added. Variables *parallel* and *bifacial* were transformed to zeros and ones instead of booleans (True or False) and removed soon after, as it is highly unlikely that one out of four systems contains bifacial modules. Missing cloud coverage values were substituted with mean which did not include NaNs in the sample count. Training examples containing zero GHIs were removed.

2.2.1 Daily Inputs

Correlations between features in the daily data can be seen in figure 2.6 where black rectangles correspond to strong negative correlations and white rectangles correspond to strong positive correlations. Upper and bottom parts of the heatmap, split by the diagonal, are symmetrical to each other and analysing just one of them is sufficient. The data used for plotting figure 2.6 was extracted several times throughout the project and the presented figure contains only features kept for feature selection. However, in prior versions of the heatmap it was discovered that number of PV modules and module nominal power were strongly correlated with power of the entire system, hence were removed. Ground temperature was correlated with ambient temperature and was removed as well.

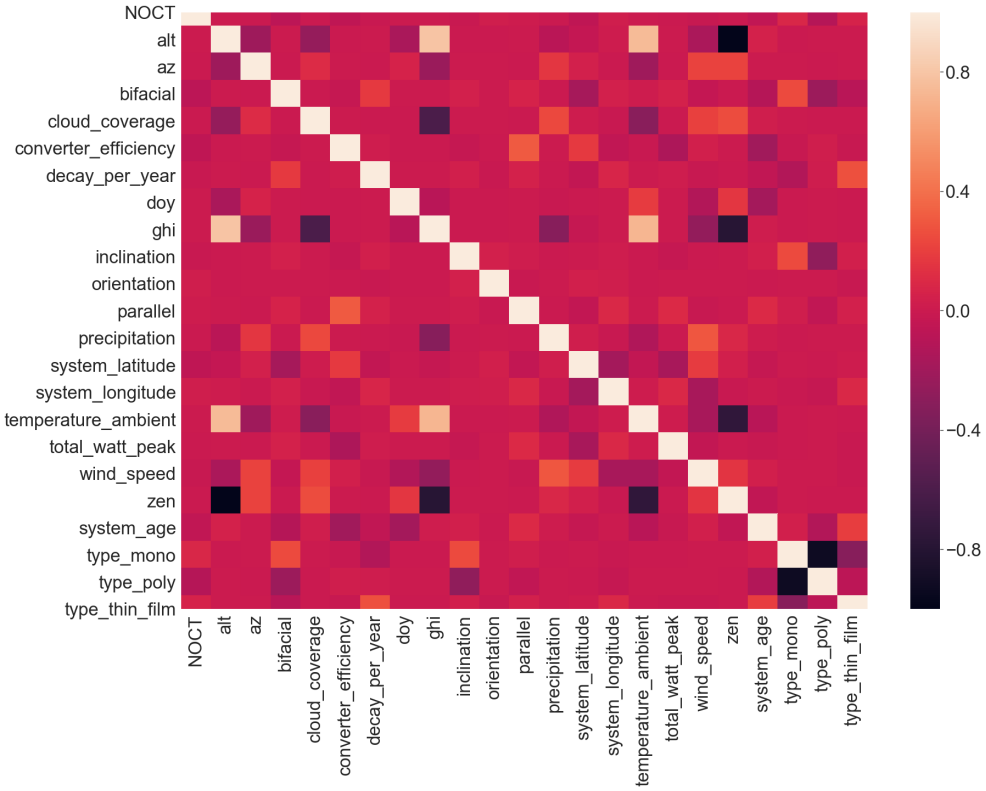


Figure 2.6: Heatmap showing correlations between all features - daily data set

In figure 2.6 it can be seen that GHI is positively correlated with sun altitude and negatively correlated with cloud coverage. That is intuitive, as clouds cause shading and reduce irradiance incident on flat surface on the ground. Sun zenith has strong

negative correlation with ambient and ground temperature, as well as with GHI and sun altitude. That is understandable, as the higher Sun zenith, the more irradiance reaches Earth's surface and the hotter it becomes. Because of these strong negative correlations solar zenith should have been removed. However, due to an oversight it was left for feature selection.

Even though day of the year might seem to encompass yield seasonal variations, initially it was not used as a feature. It was believed that predictions should be generated based on weather conditions and system parameters, not on time of the year. Rationale behind was that PV yield should be of particular value because of some specific weather conditions, rather than because it is July. If the same conditions occurred in February, yield forecast should be identical. However, this contradicted [Massaoudi et al., 2019] who claims there is a strong relation between generation on the same day of the year during different years (e.g. yield on April 4th in 2018 and 2019 is likely to be strongly correlated). Even though the analyzed data set contains values only for one year and could not benefit from yearly patterns, *day of the year* feature was assumed to provide a link between data for different systems and training examples. If on July 1st many systems had high yield, it is more likely that the analyzed system might have high yield as well. Therefore, the feature was left in the data set and significantly contributed to algorithms' performance (vide appendix B).

It is important to notice that only features used by analytical models were taken into account. Other parameters, such as air pressure, were neglected despite being utilized by other researchers such as [Kuzmiakova et al., 2017]. It should be also noticed that voltages and currents are not inputs to the described algorithms, as obtaining these measurements was not feasible for all 1,102 systems.

2.2.2 Hourly Rough Inputs

Daily and hourly rough data sets utilize the same set of features with the data resolution being the only difference. Correlations in this data set are presented in figure 2.7. This data set was subject to similar processing techniques as previously described for the daily data.

2.2.3 Hourly Detailed Inputs

The last data set contains GPOA pre-calculated for each module and averaged to obtain per system values. For each module its inclination, orientation, latitude and longitude together with GHI and solar position are used to calculate GPOA. Solar position is calculated based on solar azimuth and altitude. All mentioned values are used to calculate direct, diffuse and reflected components of light reaching each module. Obstacle factor, which is calculated using 360° maps of the surroundings, was utilized to correct the direct component of light as can be seen in figure 2.8. Two versions of GPOA calculation, with and without obstacle factor, were tested. It was discovered that using GPOA, which already included shading, worsened performance of all models. Flowchart of the obstacle factor calculation can be seen in figure 2.9. Due to a glitch in Solar Monkey's database extraction of all inputs necessary for individual albedo calculation was not possible. Therefore, an assumption of constant albedo of 0.2 was used (dr H. Ziar, personal communication, December 6th, 2019). Next, mean values for all modules were calculated. Performing calculations per module could provide better performance than per-system approach, but was not feasible due to lack of per-module yield data.

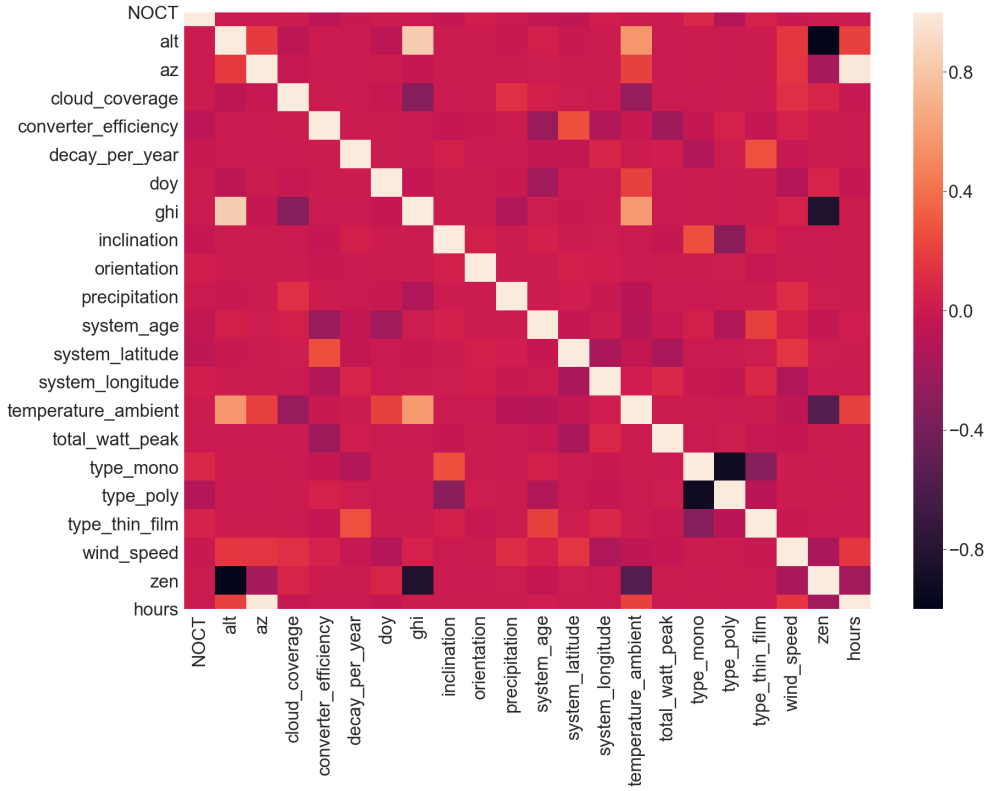


Figure 2.7: Heatmap showing correlations between all features - hourly data set

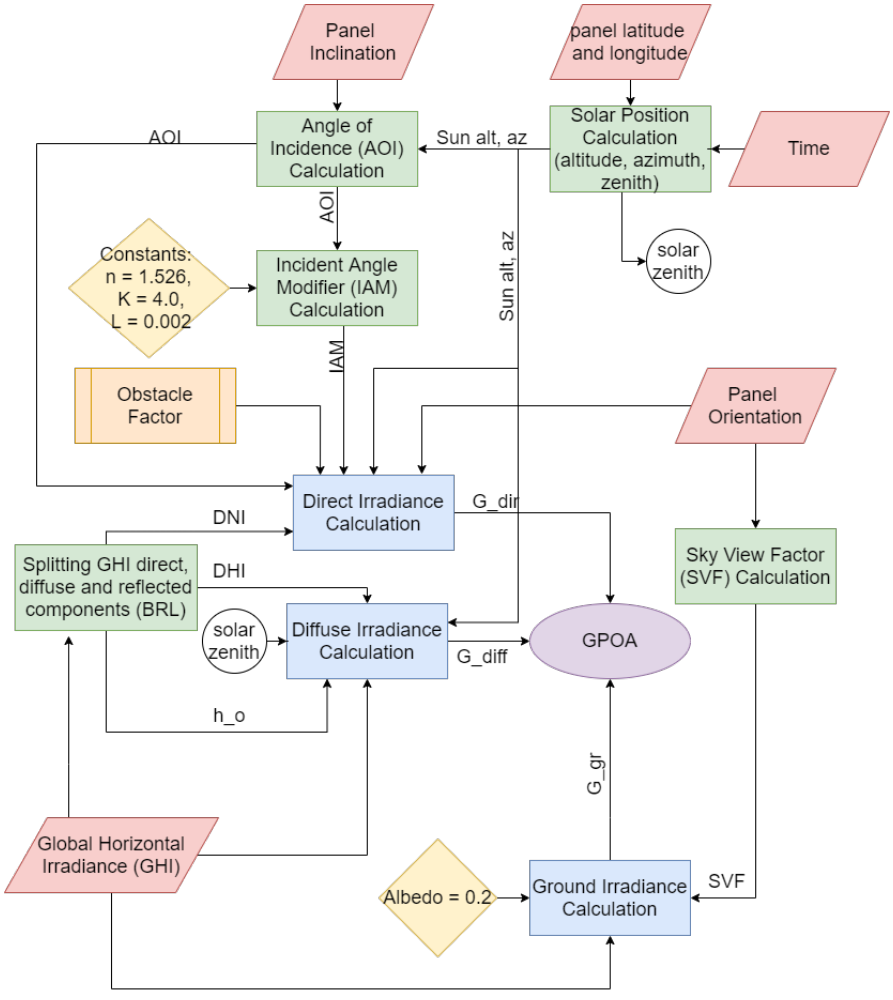


Figure 2.8: Flowchart of GPOA calculation - hourly detailed data set

It is important to stress that the code used to generate this results was not created entirely in this project, but is based on the work of Alba Alcaniz Moya and Annanta Kaul, Solar Monkey interns in 2019 [Solar Monkey, 2019a], [Solar Monkey, 2019b]. Detailed description of utilized equations can be found in summer internship reports and [Smets et al., 2016]. Remaining features were pre-processed as described before.

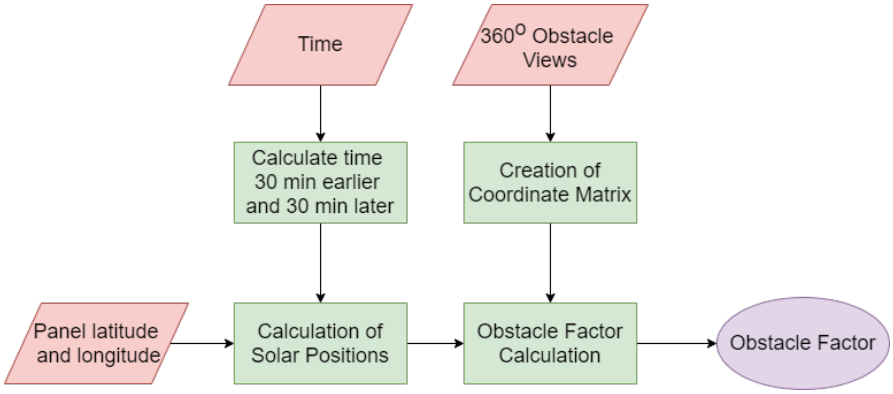


Figure 2.9: Flowchart of obstacle factor calculation - hourly detailed data set

Heatmap created for the data set containing GPOA calculations did not differ significantly from figure 2.7.

2.3 EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis (EDA) is an important step during which the data set is examined to identify potential data quality issues, calculate variance and create visualizations. It was performed separately for daily and hourly data, however large overlap exists. Therefore, common description is provided first and then differences are addressed in subsections. Interesting guidelines for EDA can be found in [Shin, 2020].

During EDA several interesting discoveries were made. Feature *parallel* turned out to have zero variance and was removed. According to another feature, 22.37 % of all modules were bifacial. That seemed suspicious, as bifacial technology is relatively new and did not have the time to disseminate in one quarter of the market. Based on (J. Donker, personal communication, January 5th, 2020) it was determined that this feature has misleading name in Solar Monkey database and in reality corresponds to type of PV power optimizer. Therefore it was removed as well. All data sets contain the same system ids, hence distribution of module types is similar in all cases which can be found in figure 2.10. No correlation between the systems age and their size was observed. Systems have sizes varying between 1.59 kWp and 17.7 kWp. Their distribution can be seen in figure 2.11. This indicates that this research focuses on small-scale residential systems. Due to the chosen loss function models tend to focus on large systems. In order to prevent low quality predictions for small systems, data for installations with total installed power above 17.7 kWp were removed. Distribution of the systems age and a scatter plot with respect to their size can be found in figures 2.12 and 2.13, respectively. Clearly, no correlation between systems size and their age exists.

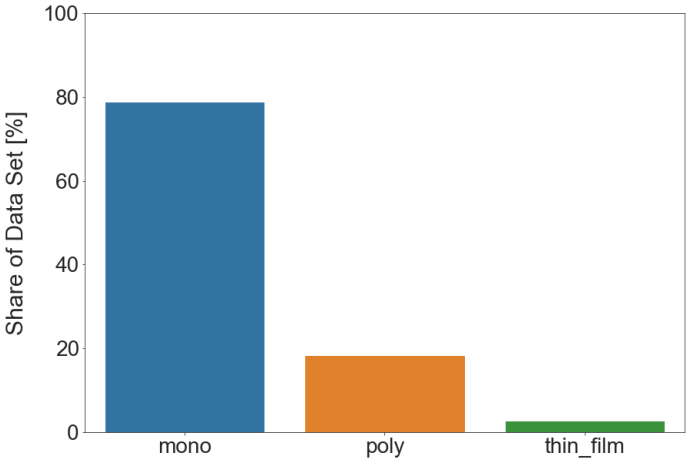


Figure 2.10: Module type histogram

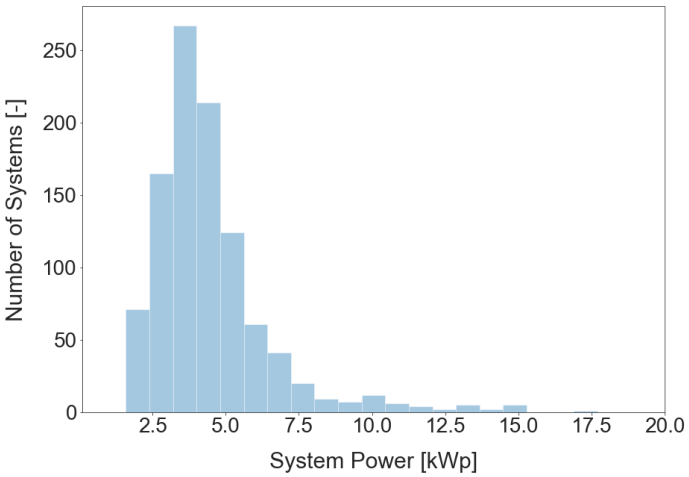


Figure 2.11: System size histogram

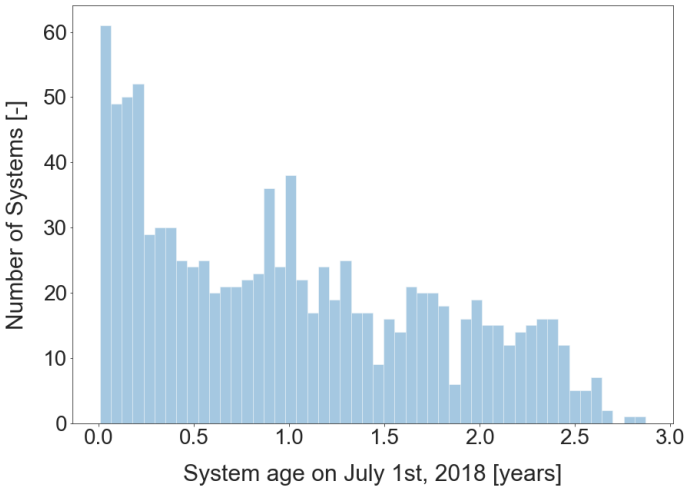


Figure 2.12: System age histogram

For all the data sets the histogram of wind speeds, visible in figure 2.14, resembles Weibull distribution [Bowden et al., 98], therefore it is assumed to be reliable. Distribution of ambient temperature for both daily and hourly data is similar and can be seen in figure 2.15.

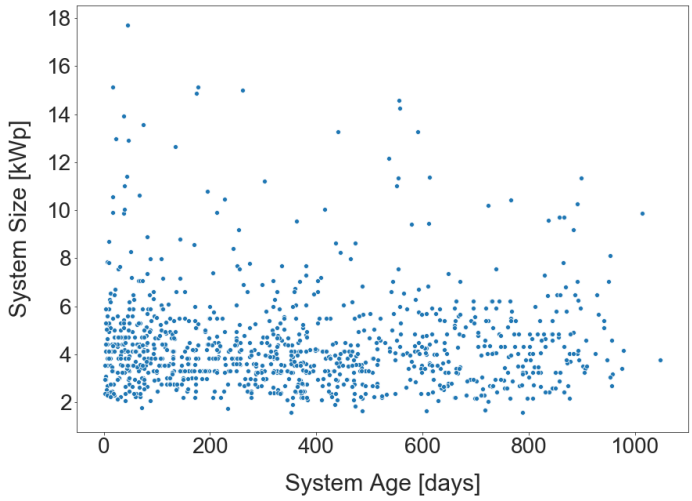


Figure 2.13: System age vs. system size

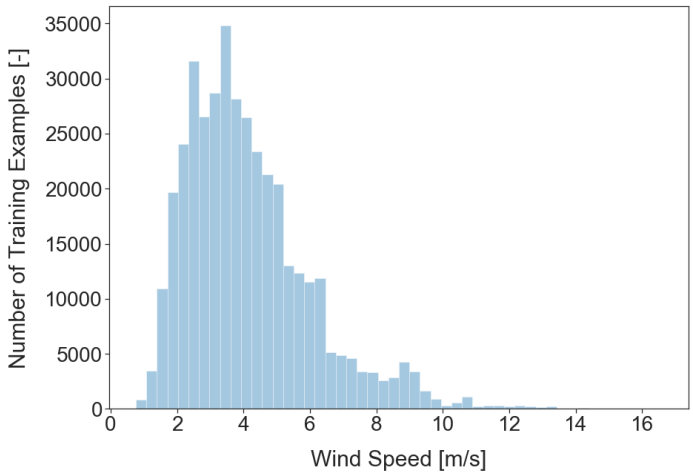


Figure 2.14: Wind speed histogram - daily data set

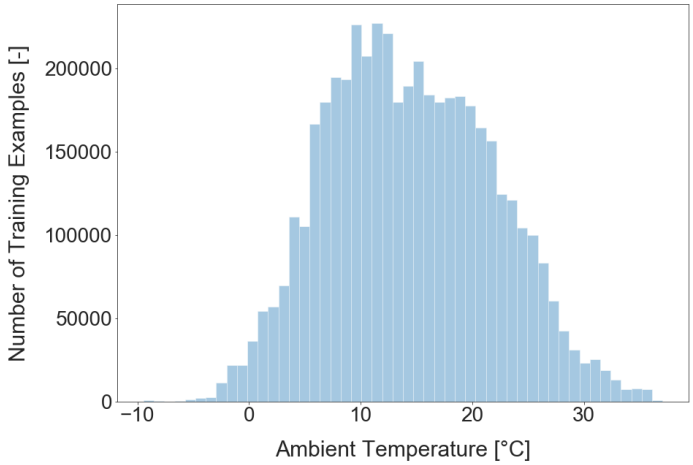


Figure 2.15: Ambient temperature histogram for hourly data

Analyzed systems are younger than three years, usually do not have size exceeding 7.5 kWp and consist mostly of mono-crystalline modules. Weather data regarding wind speeds and ambient temperature do not present any aberrations.

Discrepancy between cloud and irradiance data exists and is described in the next section.

2.3.1 Daily Inputs

Before irradiance and cloud distributions are presented, short numerical overview of daily weather data is provided. Ambient temperatures remain between 268.3 K (-4.85 °C) and 305.9 K (32.7 °C) which is reasonable. Rainfall is present in 13.4% of all training examples and its maximal value equals 2.9 mm per day. GHI in figure 2.16 is relatively evenly distributed except the peak around 50 W/m^2. None of the values exceeds 500 W/m^2 which could seem suspicious if it was not a daily mean.

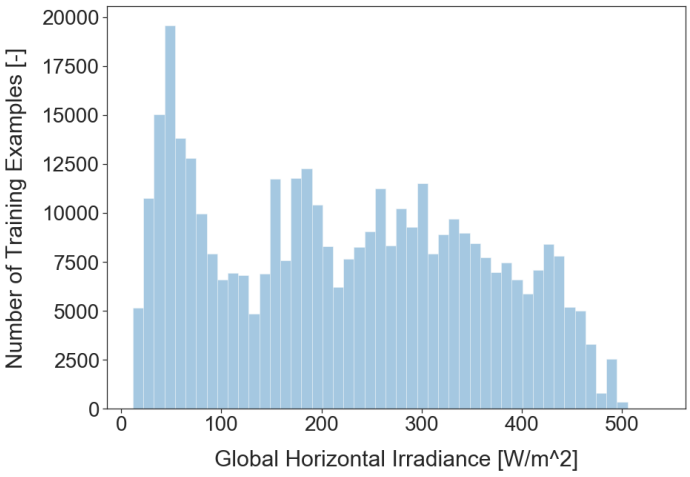


Figure 2.16: GHI histogram - daily data set

The histogram of cloud coverage data (figure 2.17) revealed that a vast majority of training examples have values corresponding to completely overcast sky. In Dutch climate that is not very reliable. However, other source of cloud coverage data was not available.

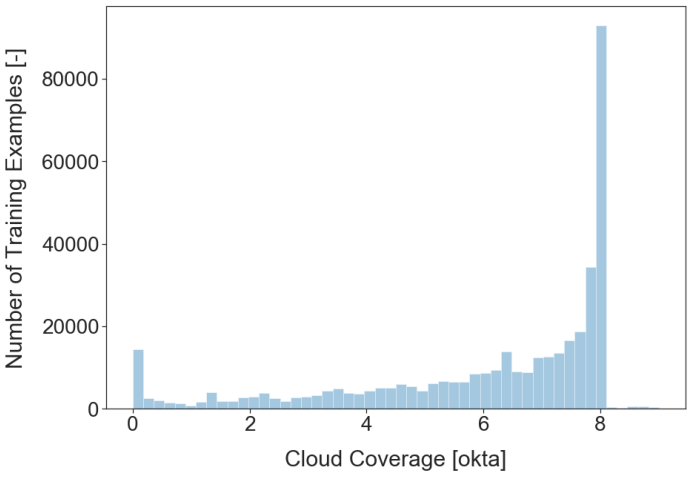


Figure 2.17: Cloud coverage histogram - daily data

2.3.2 Hourly Inputs

The most explicit difference between EDA performed for the two data resolutions is visible when figures 2.16 and 2.18 are compared. Although hourly GHI never

exceeds 965 W/m^2 which is typical in the Dutch climate, its most frequent values are in the smallest range. The reason for that might be that GHI is often decreased due to clouds and, therefore, its values rarely are close to Standard Test Conditions (STC). Small yields occur twice more often than large ones, as usually there is only one irradiance peak around midday and two moments of low irradiance, after sunrise and before sunset. This justifies the shape visible in figure 2.18.

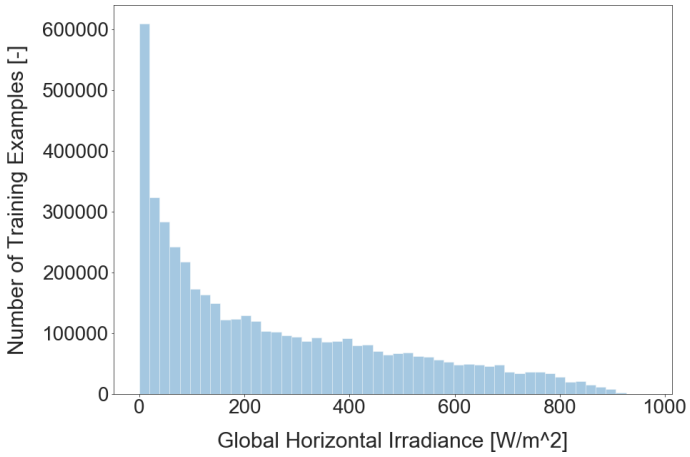


Figure 2.18: GHI histogram - hourly data set

Distribution of hourly cloud coverage data raises similar concerns as its daily counterpart. Distribution in figure 2.19 is even more explicitly skewed towards extremes - completely clear and completely overcast skies. That again raises suspicion regarding cloud coverage data quality. In hourly data set ambient temperature re-

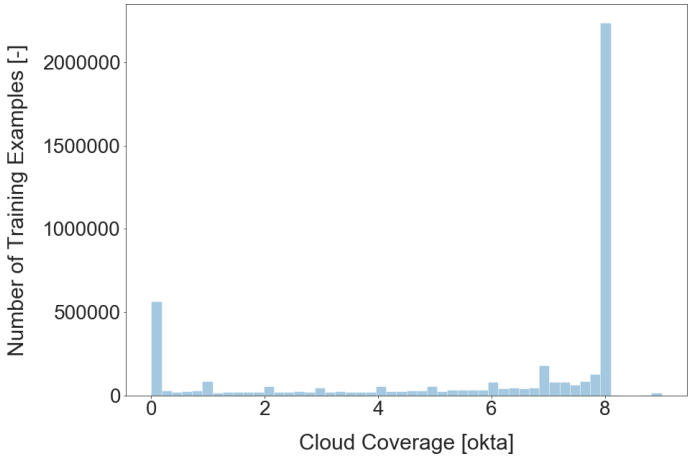


Figure 2.19: Cloud coverage histogram - hourly data

mains between 264.2 K (-9 °C) and 310.2 K (37.05 °C) which is reasonable. Rainfall is present in 13.4% of all training examples and its maximal value equals 21.6 mm per hour or 37.14 mm per day. The latter is significantly different than for hourly data and can be justified by daily data set using mean daily precipitation values.

2.3.3 Future Time Steps

In order to evaluate the best model performance for individual systems and determine whether it scales well to previously unseen systems, data for 1,102 PV systems for time steps between July 1st, 2019 and May 30th, 2020 were extracted. It was performed only for the hourly rough data set.

2.4 DATA SCALING

Another step of data preprocessing is *standarization* used to obtain similar magnitudes of all features. This is required by ML algorithms to provide optimal performance. Equation 2.1 presents the working principle of standarization where μ is the mean of all instances of particular training feature, σ is their standard deviation, and z is the standarized value of the given sample x [Raschka, 2014].

$$z = \frac{x - \mu}{\sigma} \tag{2.1}$$

Instead of the simplified version (equation 2.1), each of the data sets was standarized using sklearn *MinMaxScaler* described by equation 2.2, where x is a sample vector, x_{min} and x_{max} respectively correspond to the vector minimum and maximum, max and min correspond to feature range. Finally, x_{norm} is the outcome, a single standarized value.

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}(max - min) + min \tag{2.2}$$

Min-max scaler preserves the original shape of the data distribution and is a standard choice. Use of *Robust Scaler* is recommended if it is required to reduce the influence of outliers [Pedregosa et al., 2011]. This is not the case, as all outliers were removed. There is no need to obtain normally distributed outputs, hence *Standard Scaler* neither was used. Another alternative to Min-Max scaler is row-wise normalization, contrary to column-wise standarization. Normalization re-scales values into range [0,1] and is particularly useful when all parameters need to have positive scale. However, the outliers of the data set are lost and for most applications standarization is recommended. A common mistake is to standarize data before splitting them into training, validation, and test sets which means that the mean and the standard deviation are calculated on all samples. This leads to data leakage, as the test set is no longer a *left-out set*. More information about rationale behind choosing the right scaler can be found in [Hale, 2019] and [Dorpe, 2018]. Detailed and relevant information about the process can be found also in sklearn documentation [Pedregosa et al., 2011].

2.5 DIMENSIONALITY REDUCTION

Each feature of a data set adds to its dimension. In order to represent them graphically or to decrease computational time of the models, PCA can be used. It is an unsupervised machine learning technique which allows to decrease data set size at the expense of its variance [Bishop, 2006]. It is reasonable to ensure that at least 99% of variance is maintained, otherwise useful information contributing to the model performance can be lost. Several experiments utilizing PCA were performed, but loss of variance significantly harmed the performance of developed models while the computational time remained high (threshold of 99% of variance was used). For this reason PCA usage was neglected, but might be worth investigating in the future. PCA is used after data scaling and before feature selection. Its working principle and underlying theory can be found in [Bishop, 2006].

2.6 FEATURE SELECTION

Feature selection is a process of eliminating features which do not contribute to increasing algorithm's performance. It is one of the most important steps and has paramount importance on model outcomes. In this section a variety of feature

selection methods are presented. However, the list of techniques is by no means exhaustive. More information can be found in [Li et al., 2020] and [Agarwal, 2019].

Default *sklearn* feature selection functions are: *K-Best*, *Variance Threshold*, *Select From Model*, and *Recursive Feature Elimination (RFE)* with or without cross-validation. The aim of this process was to find the optimal configuration both in terms of performance and computational time. *K-Best* allows selection of an arbitrary number of features and removes the rest. Similarly, *Variance Threshold* method eliminates features having variance below a selected value. Next, *Select From Model* allows to choose a number of features with the highest weights calculated by a machine learning model. All of these methods except RFE require providing an arbitrary number as input and can find optimal feature configuration only by trial end error. Taking into account the number of data sets and models it is inefficient to repeat this process for all of them. Therefore, *K-Best*, *Variance Threshold* and *Select From Model* techniques were neglected. The main objective of feature selection in this project was specifying the number of features worth retaining, rather than setting it blindly. Additionally, the obtained result should be independent of shuffling. The technique meeting these requirements is RFE with 3-fold cross validation.

RFE calculates the chosen metric for algorithm trained on all features. Next, it drops feature with the lowest importance and recalculates the metric. This process is repeated until discovering the optimum. Based on the metric values, features are sorted from the most to the least influential and split into categories. Parameters detrimental for overall performance are removed from the input matrix. Before dropping a feature, training process is repeated for 3-times and the mean metric of all runs is calculated. This way it is ensured that picking different data for training and evaluation would provide similar outcome. It is important to notice that results highly depend on the model used as input to RFE function. Using the same model for feature selection, as for making predictions does not always lead to the optimal solution. For example, in case of polynomial regression, using it for recursive feature elimination provided much worse predictions than the use of random forest. That is a frequent situation in the machine learning which is a highly empirical field and in some cases trial and error is inevitable. It is worth noticing that random forest is often used for feature selection because of its unique working principle.

Additional benefit from feature selection is shortening the computational time and making the implementation easier. When using fewer features, less data has to be extracted from Solar Monkey's data base and plugged-in to the models. Feature selection is performed only once during research and does not have to be repeated after model implementation.

Several algorithms including XGBoost and random forest can also be used to sort out the features from the most to the least useful. An approach based on combining weights discovered by several algorithms was pursued by [Massaoudi et al., 2019] who created *partition vector importance*. They used three different algorithms which assigned different weights to the analyzed features. Then, another weight was assigned to each algorithm and several arbitrary scenarios were analyzed. This approach was not pursued here to prevent excessive model complexity and high computational burden.

3 MACHINE LEARNING KEY CONCEPTS

This chapter aims to provide the ML background for the readers with no prior experience in this field. All described procedures assume that data has been already cleaned and is provided in two aligned matrices, as described in chapter 2. Theory provided in this chapter is of paramount importance for making an educated decision about the next algorithm to try. Understanding these concepts is also necessary for the analysis of the best model in chapter 5.

3.1 DATA SPLITTING

In order to obtain reliable results, models have to be evaluated on previously unseen data, so called *left out set*. Therefore, all available data set is split into training, validation and test sets. Training set is used for model fitting, validation set is used for adjusting its parameters (*tuning*) and the test set is used for model evaluation only. Utilization of the validation set is necessary, as tuning algorithm parameters on the test set would optimize the model towards it and lead to over-optimistic results, so called *data leakage*. Usual train-validation-test ratio is 3:1:1, but with large data sets 98:1:1 can be used [Ng, 2020]. That is because as much data as possible is used for the training purposes and 1% of large data set still provides sufficient number of samples for the model evaluation. However, according to the learning curves which can be found in chapter 4 and appendix D none of the models would significantly increase its performance when trained on more data. Other reasons to stick with the old fashioned 3:1:1 split are shorter computational times and higher validation confidence.

3.2 CROSS-VALIDATION

Train-validation-test split is dependent on the picked samples. It might happen that one of the subsets contains "easier" or "harder" configuration of samples to predict, therefore skewing the obtained results. In order to prevent this situation *cross-validation* is used and its working principle is depicted in figure 3.1. Blue squares represent parts of the data set used for model testing while grey rectangles represent the training set. 10-fold cross validation splits the entire data set into 10 pieces meaning that always 90% of it is used for training and 10% of it is used for testing. At each iteration model is evaluated on the left-out fold and after this process is done for all folds, the mean of all results is calculated. The main advantage of this method is its independence of shuffling. In this project 3-fold cross-validation was utilized. Default cross-validation used to extract results described in chapter 4 splits training set randomly. It treats all samples independently and shuffles them in a random manner focusing only on maintaining the specified ratio between the training and testing data. This way data sets have uneven number of training examples corresponding to each PV system. In case of the daily data, it might happen that for one system e.g. 200 out of 365 samples are in the test set. For another system this could be fewer samples, hence daily metrics for both systems could not be compared. Moreover, larger number of training examples used for testing implies that fewer examples are used for training which could

Figure 3.1: Working principle of 10-fold cross validation

result in worse model performance for under-represented systems. A solution to this problem is splitting data for each system with a constant ratio among training and testing folds and maintaining it during cross-validation. Instead of treating all inputs as uniform, samples for each PV system would be extracted and split into five folds. This way all systems would have sufficient number of training examples and the test set metric calculations would remain reliable. This method is called *PredefinedSplit* and is depicted in figure 3.2. Pink rectangles correspond to the testing folds and the white rectangles correspond to the training folds. Each fold is used for testing once. Similar is true for figure 3.3. Alternatively, data set splitting can be

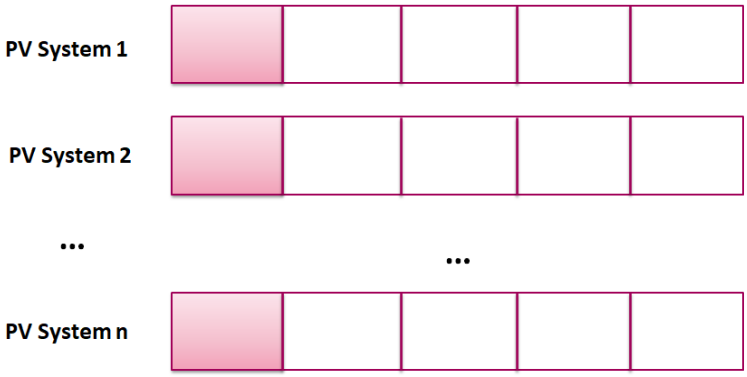


Figure 3.2: Predefined split cross-validation

performed with respect to system ids. All data for some PV systems might be used for training while other systems are used entirely for validation or testing. This way a situation when model was trained on existing systems and is used to make predictions for new (previously unseen) systems can be examined. This split is called *GroupShuffleSplit* and its graphical representation can be seen in figure 3.3. During every split different combinations of systems are used for training and testing. The relevance and implications of *GroupShuffleSplit* are explained in chapter 5. In this section the working principle of cross-validation was presented. It was explained that its main purpose is making results independent of shuffling. More advanced methods of cross-validation, such as *GroupShuffleSplit* and *PredefinedSplit* were presented and the specific issues tackled by each of them were described. Thorough understanding of these techniques is necessary for preventing data leakage.

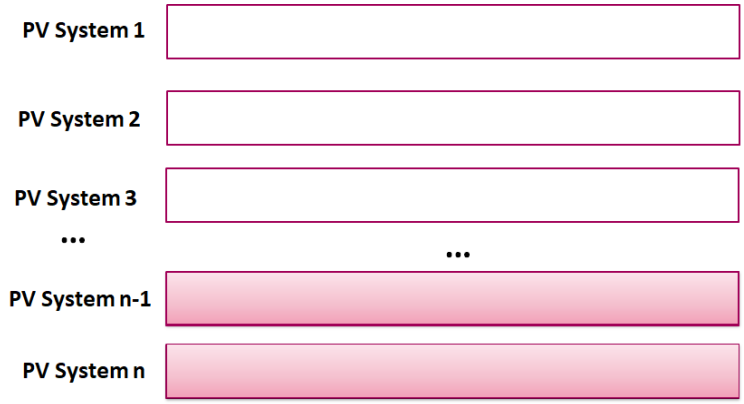


Figure 3.3: Unseen systems split cross-validation

3.3 COST FUNCTIONS AND LEARNING RATES

The process of algorithm training is based on minimizing the equation called *cost function* which calculates error between predictions and observations. The method of performing minimization depends on the chosen optimization algorithm, a popular choice is *gradient descent* described by formula 3.1.

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \tag{3.1}$$

Gradient descent is used to iteratively update model parameters and the size of its steps is called *learning rate*. Choosing the optimal value of this hyperparameter is a tradeoff between the computational time and the overshooting the global minimum. The smaller the learning rate the higher model performance and the longer computational time. On the other hand, too large values of the learning rate can cause convergence failure, as the algorithm would keep overshooting the optimum. Cost functions differ depending on the kind of utilized algorithms. One of the simplest possible choices is the *squared error loss* described by equation 3.2

$$\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - \sum_{j=0}^{p} w_j * x_{ij})^2 \tag{3.2}$$

3.4 BIAS VS. VARIANCE TRADE-OFF

Machine learning algorithms suffer from two main sources of error, namely *high bias* and *high variance*. If a model has high error on the training set, it suffers from high bias. That means its complexity is insufficient to capture the pattern in the input data. This problem is usually solved by reducing regularization, decreasing learning rate or utilization of a more complex non-linear algorithm. *Regularization* reduces the magnitude of weights found by a machine learning algorithm. Its main goal is to help the algorithm focus on the pattern while ignoring the noise.

Model having low error on the training set, but suffering from high error on validation set has high variance. In other words it *overfits*. That means algorithm fits too closely to the training data instead of following the general pattern and fails to generalize to new examples. Ways of fixing overfitting are reducing regularization and reducing number of features. Both high bias and high variance situations can be seen in figure 3.4. High bias (underfit) can be seen on the left, correct model is in the middle and high variance (overfit) is on the right. Ideally, models should have low bias and low variance meaning they have low error on the training set and

generalize well to previously unseen data. An important tool which helps to choose

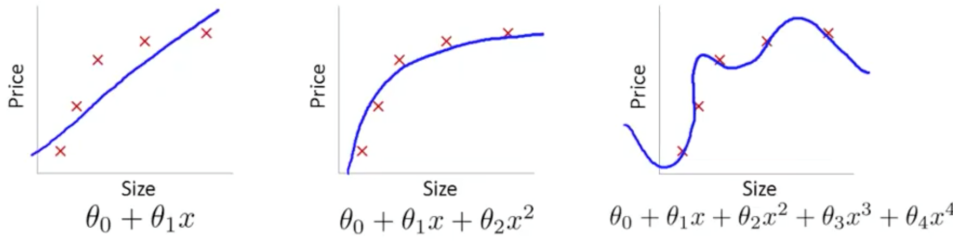


Figure 3.4: Bias vs. variance trade-off

sensible next step is the *learning curve* depicting the model train and validation errors as a function of data set size. Determining whether created model suffers from high bias or high variance is crucial for its tuning. Separate parameters influence bias and separate parameters influence variance. In case of high bias acquiring larger amount of data will not improve the model performance. Bias vs. variance trade-off is also of paramount importance for determining which algorithm should be tried next. In order to understand it figure 3.5 can be analyzed. In figure 3.5

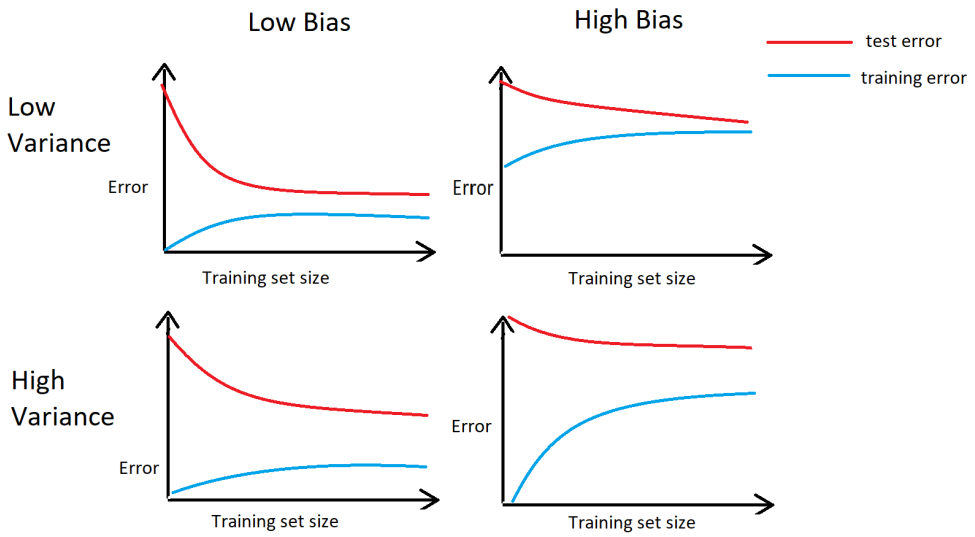


Figure 3.5: Bias vs. variance - learning curves

blue lines correspond to the training errors and red lines correspond to the test errors. With increasing training set size training error increases, as it becomes harder to fit regression line to all samples. At the same time testing error decreases, as a model trained on larger number of samples performs better. High bias means that a model has high error. Bias is expressed as vertical distance between x axis and the error line. Variance is expressed as vertical distance between the training and testing curves. Models with high variance fit closely to the training data, but fail to generalize to new examples. Therefore, for models suffering from high variance, the vertical distance between training and testing error curves is large.

One more important aspect of bias vs. variance trade-off is it is used for determining model accuracy and precision. **In ML there is no single metric corresponding to these metrics**. In case of classification accuracy and precision are very popular, as they quickly provide intuitive insight into a model's performance. Accuracy corresponds to how often predictions match the reality and precision defines the

spread of error. Learning curves allow to determine whether model is accurate and precise, as depicted in figure 3.6 complementary to figure 3.5.

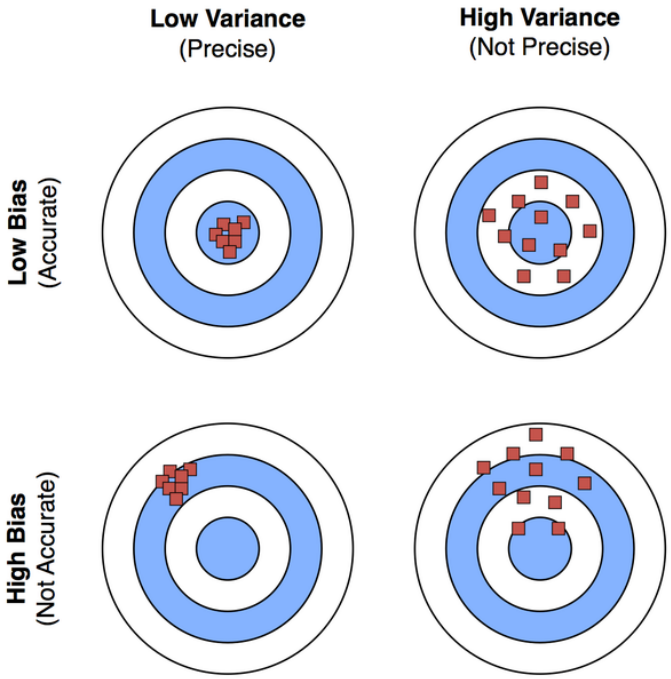


Figure 3.6: Accuracy and precision for regression

As described before, models with high bias are not complex enough to capture pattern in the data, hence they are unable to make predictions close to observations (low accuracy). Models with high variance fit to closely to the training set, but fail to generalize to new examples. They suffer from low precision. An ideal ML model should have low bias and low variance which means it should be both accurate and precise.

3.5 HYPERPARAMETER TUNING

Hyperparameter tuning is an iterative process of finding the optimal configuration of algorithm settings and can be performed either manually or automatically. Hyperparameters differ among models, but despite their huge variety, most of them in different ways address bias vs. variance trade-off described in the previous section. The higher the model complexity, the larger the number of adjustable hyperparameters.

Tuning can be performed manually by calculating the chosen metric on training and validation sets. By determining whether the analyzed model suffers from high bias or high variance, a better configuration can be found and tested. However, this process is tedious and time consuming. Alternative method would be *GridSearch* which tries all configurations for the given hyperparameter lists. A matrix containing the calculated metric for each possible configuration of the given hyperparameters is created. This method guarantees finding the optimal solution, but can be very time consuming for complex algorithms and large data sets. Additionally, it might remain in sub-optimal region for a very long time. Its most popular alternative is the *RandomizedSearch* which tests only an arbitrary number of hyperparameter configurations within specified ranges. If the number of samples is sufficient, a solution close to the optimum can be found in relatively short time. It is important to no-

tice that both *GridSearch* and *RandomizedSearch* should be cross-validated in order to provide results independent of shuffling.

3.6 METRICS

Model validation is a crucial step in the entire modeling process. It is based on metric calculation which allows comparison of different models. Not only is it used at the last stage of modeling, but also for algorithm training. After each training iteration the chosen metric is calculated and compared with the previous result. Based on the comparison algorithm can tell whether last change of hyperparameters was beneficial or not and make new changes accordingly. In this section formulas and descriptions of all utilized metrics are provided. Metrics discussion is crucial for solar yield nowcasting real-life applications. Creating one model for multiple systems requires a metric which combines small absolute error with treating all systems equally. As discovered in literature study, the most commonly used metrics are MAE, RMSE and MAPE. An overview of metrics used in the selected studies can be found in appendix A.

Mean Bias Error Mean Bias Error (MBE) is described by equation 3.3 and expressed in kWh. It is not a good candidate for the metric, as it allows positive and negative errors to cancel out. Please note that in all equations in this section $\hat{y}$ corresponds to the prediction vector, y corresponds to the observation vector and n corresponds to the total number of observations.

$$MBE = \frac{1}{n}\sum_{i=1}^{n} y_i - \hat{y}_i \tag{3.3}$$

R^2 called *adjusted coefficient of determination* can be calculated using equation 3.4 where $\overline{y}$ is the mean of all predictions. It is one of the most popular metrics for regression models. Its maximal value is one, corresponding to a perfect model, and it has no minimum, as a model can be infinitely wrong. R^2 can be also interpreted as a probability that the next prediction point will occur directly on the regression line. This metric is very useful in the initial research phase when models are relatively bad and tend to have metric values far from 1. As R^2 tends to be too high, when the models improve, it does not alter enough to depict relevant changes. Additionally, it does not capture overfitting and tends to be exaggerated when calculated on time series data.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2} \tag{3.4}$$

Mean Absolute Error (MAE) is expressed in kWh and calculated using equation 3.5. MAE does not include system size, therefore tends to be large for large systems. Also, it does not penalize extreme errors additionally, contrary e.g. to Mean Squared Error (MSE).

$$MAE = \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{n} \tag{3.5}$$

Mean Absolute Percentage Error (MAPE) seemed to be perfect for the assessment of PV yield monitoring quality, as it is relative to system size. However, percentage errors might take high values for low absolute yield mismatch which takes MAPE far from the desired range of 0 to 100%. For example measurement of 0.5 kWh and prediction of 0.1 kWh would generate percentage error equal to 400%. This would skew MAPE which is mean of all percentage errors, as can be seen in equation 3.6. Such high value indicates large error, but does not inform it is of little importance.

This situation is particularly noticeable for XGBoost algorithm which often has high percentage error for small values of yield.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}|\frac{y_i - \hat{y}_i}{y_i}| \tag{3.6}$$

Mean Squared Error (MSE) is expressed in kWh^2 and calculated using equation 3.7. It focuses on large systems with large error, therefore models which use it for training tend to have smaller percentage error for large systems.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.7}$$

RMSE is described by equation 3.8. Similarly to MSE it penalizes large errors more, therefore preferring multiple small mistakes over few large ones. That is essential, if the model is to be trusted. Root ensures that metric values are relatively small and easily comparable. RMSE has a drawback of being large for large systems. However, due to its extreme error handling and relatively small magnitude it is one of the most important validation tools in this project.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3.8}$$

Normalized Root Mean Squared Error (NRMSE) is the most suitable metric for the analyzed problem and can be calculated using equation 3.9. As few researchers analyzed data from large number of PV systems, they did not face the issue of model fitting focusing on the large systems. NRMSE provides scaling with system size and ensures that percentage errors for systems of all sizes remain unbiased.

$$NRMSE = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}}{max(\hat{y}) - min(\hat{y})} \tag{3.9}$$

Metric allowing comparison with other research is *skill score* described by equation 3.10 and expressed in %. It provides relative comparison with persistence model.

$$SS = (1 - \frac{RMSE_{proposed}}{RMSE_{reference}}) * 100 \tag{3.10}$$

Furthermore, metrics E_{10}, E_{50}, E_{100} and E_{500} providing results in % and expressed by equations 3.12 and 3.11 were used. In equation 3.12 p can be either 10, 50, 100 or 500, stands for *power* and sets the threshold of absolute error. Metric E_{10} gives a percentage of predictions with absolute error below 10 Wh. Similarly E_{50} gives a percentage of predictions with absolute error below 50 Wh and so on. This metric was calculated for easy comparison of the created models with the research conducted by [Visser, 2018].

$$f(\hat{y}_i, y_i) = \begin{cases} 1 & \text{if } \hat{y}_i - y_i >= p \\ 0 & \text{if } \hat{y}_i - y_i < p \end{cases} \tag{3.11}$$

$$E_p = 100\% * \frac{1}{n}\sum_{i=1}^{n} f(\hat{y}_i, y_i) \tag{3.12}$$

Lastly, custom metric created for Solar Monkey's internal use is financial compensation C expressed in €. Intuitively, a better algorithm is the one which requires

lower payments to the customers. However, monitoring software is not used for paying out guarantees and for this reason results of this metric were not presented in chapter 4. Compensation formula can be seen in equation 3.14. Both compensation quota and threshold are user-determined. It is important to notice that this metric should not be used for algorithm training or models comparison, but was created only to present the costs associated with model implementation. In equation 3.14 value of 0.21 corresponds to price of one kWh of energy and is expressed in €and the value of -7 % represents current compensation threshold. Solar Monkey pays to the customers only when the error of its predictions is larger than 7 %. Both electricity price and error threshold are parameters which can be optimized to simulate different business scenarios.

$$PE_i = 100\% * \frac{y_i - \hat{y}_i}{y_i} \tag{3.13}$$

$$C(\hat{y}_i, y_i) = \begin{cases} |y_i - \hat{y}_i| * 0.21 & \text{if } PE_i <= -7\% \\ 0 & \text{if } PE_i > -7\% \end{cases} \tag{3.14}$$

Compensation metric is highly dependent on the assumed settlement period which can be either daily, monthly or yearly. Although daily payouts are not convenient neither for the company, nor for its clients, it might be insightful to see differences among different time horizons.

All formulas described so far can be calculated either on the entire prediction matrix, or for each system individually. General calculation might hide cases for which the model performs badly and does not include the fact that each system belongs (usually) to a different entity.**Measuring overall model performance is informative, but insufficient to determine whether all customers receive predictions of high quality**. Therefore, each metric should be calculated per system and stored in an array. Next, minimum, maximum, mean and standard deviation of each array should be calculated. This approach allows also to identify the worst performing systems and narrows the scope of error analysis. However, performing individual system calculations requires splitting data according to system ids and utilization of *scikit.learn train_test_split* function would not provide reliable results. A suitable splitting method was described in section 3.1.

To sum up, multiple evaluation metrics relevant for PV yield monitoring and forecasting were presented. In the initial research phase coefficient of determination R^2 was often used. Other supplementary metric is MAE which is easily interpretable, as it expresses by how many kWh the algorithm is usually mistaken per prediction. NRMSE seems to be the most relevant of all described methods, but for simplicity RMSE was used for most of the time. Only after finding the most promising model, complex analysis including calculation of NRMSE, E_{10}, E_{50}, E_{100} and E_{500} was performed. Compensation metric was created due to misunderstanding of Solar Monkey's business model and is not informative at the moment. Nevertheless, it was added to the created library for future use. Other metrics than the described can also be utilized to evaluate regression models, but they rarely occur in context of PV yield monitoring and forecasting.

In this chapter the key concepts of machine learning were introduced. Detailed knowledge about the working principles of all utilized ML algorithms is not required to fully understand the findings of this research. Insight into mathematics underlying ML was provided in appendix E. To summarize, it is very important to evaluate models on previously unseen data, so their performance in real life can be tested. Cross-validation is a technique making results independent of data set shuffling. Models are trained and evaluated using the given metric and its choice

is of paramount importance for the outcome of any ML related research. ML models dealing with regression do not have single metrics corresponding to accuracy and precision. Metrics have to be calculated for training and validation sets and based on comparison of these results, model "accuracy" and "precision" can be estimated. Being equipped with this knowledge, it should be possible to interpret the results in the following chapters.

4 RESULTS

In this chapter results of six different models for three data sets are presented. The described models are: ElasticNet, Polynomial Regression, Random Forest, Extreme Gradient Boosting, Solar Monkey analytical model and simple persistence model. The last two are not ML models and are provided as a benchmark. Comparative evaluation of all models was performed for each data set. This chapter provides justification for the selection of the best algorithm for further analysis in chapter 5. As discussed in chapter 3.6, RMSE is the single most informative metric. E_{10}, E_{50}, E_{100} and E_{500} were calculated for hourly data sets only to provide comparison with work performed by [Visser, 2018]. Other metrics such as MAPE were calculated to provide additional insight. For each model and data set RMSE in *kWh/day* was calculated, regardless of the input data resolution, to ensure that cross-data set comparison is possible. In this chapter dissemination of such practice is put forward, to facilitate comparison across different studies. It is also proven that skill score is an insufficient benchmark.

4.1 DAILY DATA SET

In table 4.1 comparison of all used models can be seen. Two models, Random Forest and XGBoost, have RMSE lower than currently used monitoring software. Therefore, implementation of any of these models would improve monitoring quality leading to fewer wrong predictions and decreasing errors magnitude. The latter is measured by maximal error metric which is the smallest for Random Forest. It is preferred that models make many small mistakes over few large ones. ElasticNet is a linear method and was not expected to outperform any other technique and the persistence model was used merely as a benchmark. Polynomial Regression did improve predictions with respect to ElasticNet, but did not manage to surpass Solar Monkey software. Despite having higher maximal error than Random Forest, it is the XGBoost which turned out to be the best of all models compared for daily data. It has the highest SS and R^2 and the lowest MAE, RMSE and MAPE which indicate it is much better than the persistence model, has small relative errors and rarely is very mistaken. **It is important to notice that XGBoost RMSE is almost two times lower than for current Solar Monkey monitoring software**. *sklearn* and *xgboost* libraries sometimes provide different metric results for the same hyperparameters. The differences are not large and are likely to come from different number of boosting rounds (vide appendix E). Table 4.1 cannot be compared directly with results for the two remaining data sets, as the same metrics have different units. For example, MAE calculated on daily data is expressed in kWh/day while MAE calculated on hourly data is expressed in kWh/hour. The only metric which uses the same unit in all three data sets is SS expressed in %. However, its calculation is based on persistence model RMSE which in turn does depend on data resolution. These issues make it difficult to directly compare the results across data sets. To resolve them RMSE and MAE were calculated for hourly predictions aggregated per day and their values can be found in table 4.6. E-metrics for daily data were not calculated, as thresholds of 10, 50, 100 and 500 Watts are too small for this data resolution. Calculating E-metrics with different thresholds would not provide comparison with other data sets and work by [Visser, 2018], hence for the daily data was neglected.

Table 4.1: Results comparison for all models for daily data set

	R^2	MAE	Max Error	RMSE	MAPE	SS
Solar Monkey	0.95	1.5	30.24	2.237	26.44	64.98
Persistence Method	0.62	4.157	83.47	6.389	75.77	N/A
ElasticNet	0.89	2.352	45.09	3.382	75.4	47.46
Polynomial Regression	0.93	1.896	27.52	2.753	31.98	57.23
Random Forest	0.98	0.775	20.34	1.325	9.88	79.41
Extreme Gradient Boosting	0.99	0.698 [1]	23.89	1.185 [2]	9.37	81.59 [3]

1 value obtained using xgboost library, sklearn value is: 0.704
2 value obtained using xgboost library, sklearn value is: 1.186
3 value obtained using xgboost library, sklearn value is: 81.57

4.2 HOURLY ROUGH DATA SET

The results for all models and data set containing hourly rough features can be seen in table 4.2. Random Forest and XGBoost achieved similar performance. That is not surprising, as both methods include some form of decision trees. Random Forest has lower maximal error and MAPE while XGBoost has lower R^2, RMSE and higher SS. MAE in both cases is the same. Analysis of table 4.3 revealed that Random Forest has more predictions very close to the observed values and only above 500 Wh error XGBoost starts to prevail. That justifies lower XGBoost RMSE value which rapidly increases with large errors. It can be concluded that both algorithms are comparable and can be used for developing commercial monitoring software. However, due to its supreme ability to avoid large errors, XGBoost was chosen for the further analysis. Metrics for Solar Monkey analytical model were not provided, as its current version does not support hourly data resolution. Large values of MAPE in table 4.2, far exceeding 100%, do not imply that the developed models are insufficient to predict hourly yield. As will be presented in chapter 5 large MAPE is a consequence of using squared error as loss function and is skewed by large percentage errors corresponding to small measured yields. It is important to notice that the persistence model described in table 4.2 uses values from previous hour, not previous day, hence providing much better results and leading to lower skill scores. This exposes weakness of using SS to compare findings with other researchers. Table 4.3 presents

Table 4.2: Results comparison for all models for the hourly rough data set

	R^2	MAE	Max Error	RMSE	MAPE	SS
Persistence Method	0.8	0.346	8.27	0.531	70.75	N/A
ElasticNet	0.83	0.299	32.22	0.469	11176	10.88
Polynomial Regression	0.9	0.236	30.1	0.372	10504	29.33
Random Forest	0.96	0.102	28.82	0.216	417.76	59.04
Extreme Gradient Boosting	0.97	0.102	28.9	0.206	478.01	60.9

E-metrics describing fraction of all predictions with absolute error smaller than 10, 50, 100 and 500 Wh. Only for E_{500} XGBoost is better than Random Forest, by 0.31%. This means that usually Random Forest is more precise than XGBoost, but when it is mistaken the errors are of larger magnitude, worsening the metrics calculated in table 4.2.

Table 4.3: E-metrics for hourly rough data set

	E10	E50	E100	E500
Persistence Model	10.99%	22.12%	32.85%	75.16%
ElasticNet	3.2%	15.93%	30.51%	82.1%
Polynomial Regression	4.18%	20.47%	38.18%	87.84%
Random Forest	27.43%	57.38%	72.53%	96.35%
Extreme Gradient Boosting	23.55%	54.66%	71.35%	96.66%

4.3 HOURLY DETAILED DATA SET

In table 4.4 results for the last data set are presented. Solar Monkey does not have hourly model, hence it is not included in the table. Due to the same data resolution, tables 4.2 and 4.4 can be directly compared. Surprisingly, pre-calculating GPOA worsened most metrics, except MAPE, which has improved for ElasticNet and Polynomial Regression. Trends described in the previous section remain valid, as XGBoost still has lower RMSE. Surprisingly XGBoost surpasses Random Forest in metrics it did not dominate before, that is E_{50} and E_{100} in table 4.5. Detrimental impact of GPOA calculation is discussed in chapter 6.

Table 4.4: Results comparison for all models for the hourly detailed data set

	R^2	MAE	Max Error	RMSE	MAPE	SS
ElasticNet	0.79	0.347	9.42	0.533	9506	-0.95
Polynomial Regression	0.86	0.278	7.19	0.434	4076	17.69
Random Forest	0.95	0.141	5.73	0.269	624	49.11
Extreme Gradient Boosting	0.95	0.129	5.96	0.244	540	53.71

Table 4.5: E-metrics for the hourly detailed data set

	E10	E50	E100	E500
ElasticNet	2.7%	13.5%	26.61%	77.26%
Polynomial Regression	4.36%	20.99%	37.5%	82.87%
Random Forest	21.17%	46.81%	62.61%	93.83%
Extreme Gradient Boosting	19.56%	47.25%	64.27%	94.96%

4.4 CROSS DATA SET ANALYSIS

So far the results for three separate data sets were presented and a question arises: which model is the best of all? In order to answer it, common metrics for the selected models from each data set were calculated. In table 4.6 can be seen that using data with finer resolution caused more than twofold decrease in MAE and RMSE. Precalculating GPOA has worsened model performance, but results of a model using it are still much better than results of the daily model. **It is surprising to notice that hourly persistence model outperforms both Solar Monkey model and XGBoost using daily resolutions**. That means the monitoring service can be improved three times just by improving the data resolution, even if no complex model is created. Based on the results presented so far it can be noticed that the best nowcasting model is XGBoost for hourly rough data set. Therefore, all further analysis is focused on it.

Table 4.6: Comparison of XGBoost metrics for daily aggregated results

Resolution	RMSE [kWh]	MAE [kWh]
Solar Monkey	2.237	1.5
Persistence hourly	0.742	0.475
XGBoost daily	1.185	0.698
XGBoost hourly (rough)	0.352	0.18
XGBoost hourly (detailed)	0.435	0.235

4.5 FEATURE IMPORTANCE

In this section feature importance assigned to the selected variables for the best model is presented. Weights assigned to different features by the hourly XGBoost model utilizing GHI can be seen in figure 4.1. Feature weights for version with GPOA can be seen in figure 4.2. Graphs for all other models developed in this project can be found in appendix B. The only variables directly influencing PV yield

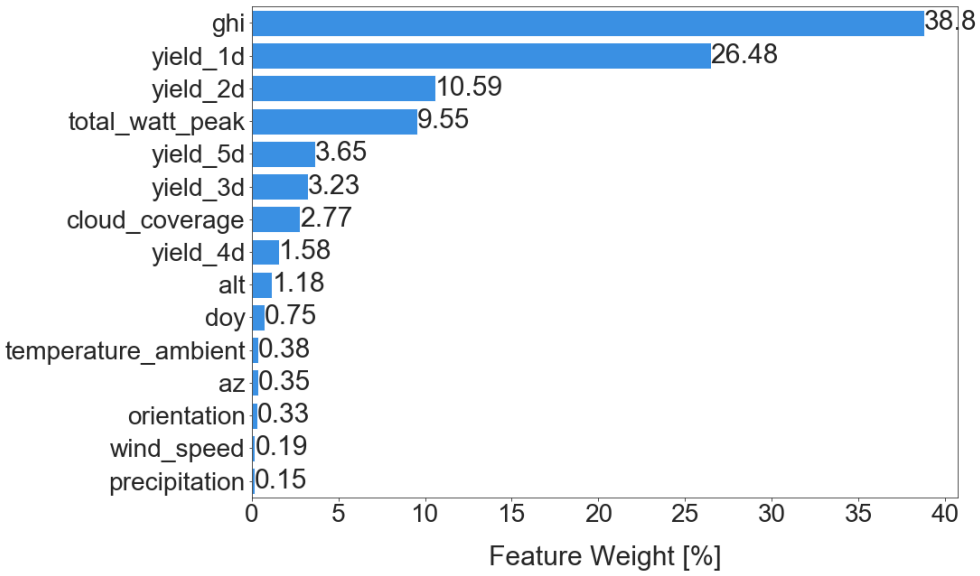


Figure 4.1: XGBoost feature importance for hourly rough data

are irradiance and module temperature. Therefore, they were expected to have the largest weights. Analyzing figure 4.1 it can be seen that GHI indeed is the single most influential feature. Total system power and cloud coverage also play an important role which is not surprising. However, it can be seen that the past yields have surprisingly high importance. This implies that the time component is very significant for the task of PV yield nowcasting. It can be also noticed that the ML models value features which have no significance for the analytical methods or assign nearly zero weights for parameters of high analytical impact. The latter is true for ambient temperature, wind speed and precipitation. Therefore, ML models cannot be used to verify existing solar engineering theory. Figure 4.2 represents XGBoost feature importance for the data set containing GPOA. It can be seen that trends described in the previous paragraph still hold with GHI replaced by GPOA as the most influencial feature.

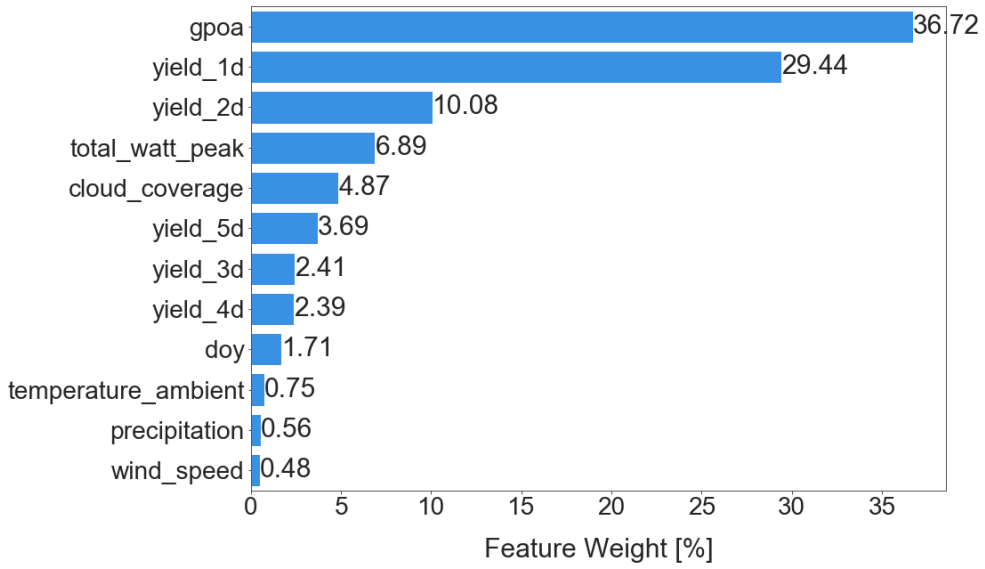


Figure 4.2: XGBoost feature importance for hourly detailed data

4.6 LEARNING CURVES

Next to the XGBoost metrics and feature importance, the selected learning curves are presented to depict bias vs. variance trade-off and learning saturation. For each algorithm there is a different training set size above which the model does not improve anymore. In this study, large data set containing more than 4,000,000 samples for 1,102 PV systems was investigated. Learning curves were obtained using 3-fold cross-validation and allow to determine whether full data set potential was utilized. Semi-transparent areas around the lines correspond to standard deviations of the results corresponding to all cross-validation folds. **It can be noticed that they are large for small data sets and decrease with increasing training set size**. That is particularly explicit when comparing figures 4.4 and 4.3 and is reasonable, as for large data sets it is less likely that particular shuffling would skew the results. Therefore, cross-validation should be neglected for the data sets larger than 0.15 % of the training set size, that is 60,000 samples (vide figure 4.4), as it significantly increases the computational cost and has no significant impact on the results.

Learning curve in figure 4.5 was plotted for custom training set sizes, which means its step value was not constant, but was adjusted manually. This approach requires repetitive plotting and adjustments, hence was pursued only for the best model. The advantage of such plotting is both initial and final stages of the model learning can be captured. Until seeing 0.15 % of the training set size XGBoost validation error decreases exponentially and it seems to stabilize later on. However, after plotting learning curve for 10 % to 80 % of training set size it can be seen that learning does not saturate and the validation error continues to decrease. However, **drop in validation RMSE is only around 0.02 kWh for the training set size between 40 % and 80 %. This supports the assumption that learning saturates**. Further increase in the training set size is likely to decrease error, but it is not justified considering surge in required computational power and the associated financial cost. Other operation, e.g. data cleaning, is likely to provide larger gain in performance at the lower expense. Next to the custom learning curve, curves depicting initial XGBoost training phase and final XGBoost training phase were depicted in figures 4.3 and 4.4. In figure 4.3 XGBoost performance converges so quickly that figure 4.4 with smaller range of x axis had to be plotted. Based on figure 4.3 it can be concluded that XGBoost has relatively low bias and variance, hence it is both accurate and precise.

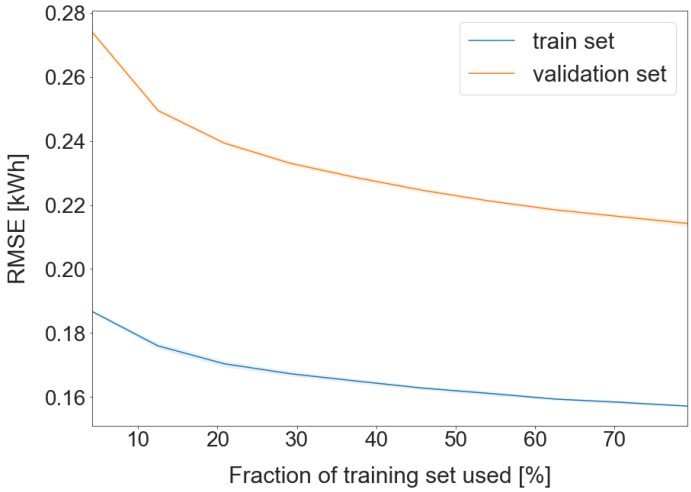


Figure 4.3: XGBoost learning curve at the beginning of the learning process - hourly rough data set

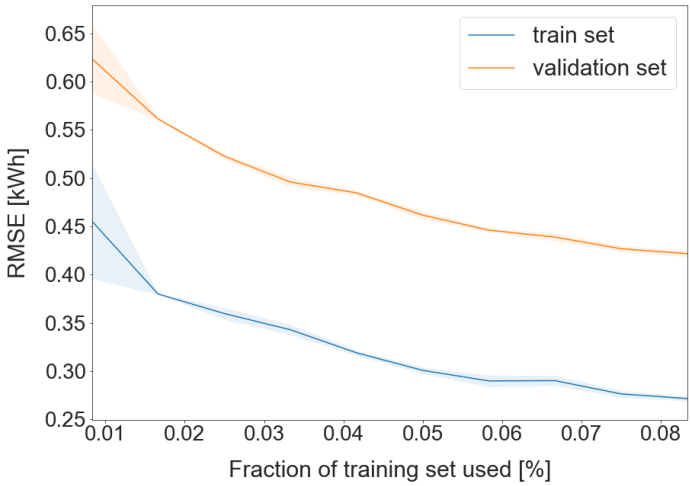


Figure 4.4: XGBoost learning curve - hourly rough data set

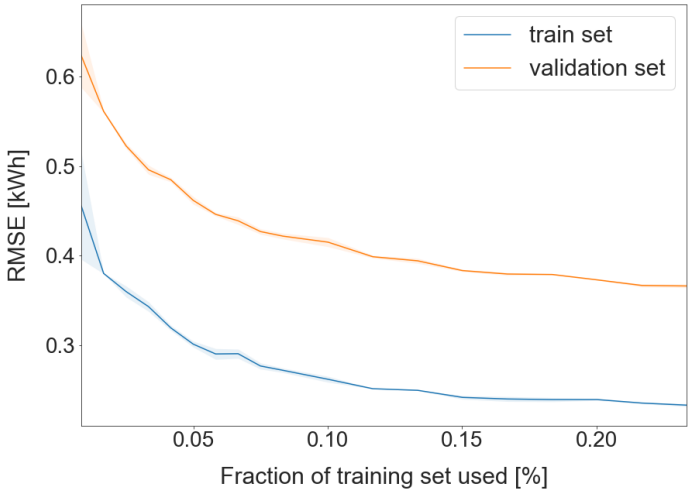


Figure 4.5: XGBoost learning curve

4.7 IDENTIFICATION OF INCORRECT STRING CONFIGURATIONS

A comparison of the top ten systems with the worst predictions for XGBoost and the analytical model was presented in tables 4.7 and 4.8. Seven IDs in bold correspond to systems which occurred in both tables and MAPE was chosen as the selection criterion. Large overlap between tables is surprising, as the analytical model and the XGBoost have completely different prediction-making process and should not be mistaken at the same time. Therefore, bad predictions are not caused by the model shortcomings, but by low quality of the input data for these systems. Comparing predictions from both models can help to identify systems with wrong string configurations (e.g. number of modules). Analyzing tables 4.7 and 4.8 it can be seen that ten the worst performing systems are of all sizes.

Table 4.7: Top 10 systems with the worst predictions by Solar Monkey

System ID	MAPE [%]	System Power [kWp]
843	1993.31	4.86
4449	1862.69	9.45
4322	1800.77	2.97
5513	1629.99	13.25
112265	1456.57	6.27
35788	1404.85	4.72
93815	626.58	3.42
39794	573.92	5.5
134296	567.26	4.72
50987	500.17	3.3
Average	1241.61	

Table 4.8: Top 10 systems with the worst predictions by the best XGBoost model

System ID	MAPE [%]	System Power [kWp]
4449	755.97	9.45
5513	659.12	13.25
112265	324.93	6.27
134296	310.84	4.72
35788	296.01	4.72
106172	153.13	3
101857	148.40	2.475
93815	141.90	3.42
39794	134.59	5.5
121519	133.65	2.32
Average	305.85	

5 ADVANCED ANALYSIS

In the previous chapter the choice between using GHI and GPOA and the rationale behind choosing the most suitable algorithm were described. Some of the assumptions made while developing models described in chapter 4 were acceptable for selecting the most promising algorithm, but are unacceptable in commercial applications. Making these simplified assumptions prevented development of custom solutions for low-performing models. The models presented so far were trained on randomly shuffled data with systems having uneven number of samples in the test set. As a result, the quality of predictions for individual PV systems could not be determined and the comparison of individual system metrics was impossible. Solar Monkey customers are not interested in the general model performance, but expect that their own predictions are of high quality. Therefore, in this chapter methods of measuring individual system performance are presented. Detailed analysis of several models utilizing the best performing algorithm, XGBoost, for hourly data set with GHI, is presented. An attempt to ensure that yield predictions for all PV systems are of similarly high quality is described. It is also explained under what weather conditions and for which PV systems the XGBoost is mistaken the most. This chapter focuses on the XGBoost hourly model, although most of the described issues are related to the daily XGBoost model as well.

5.1 PREVIOUS MODEL

Firstly, the evaluation of the best model described in chapter 4 was performed on the data corresponding to future time steps to simulate its real life behavior and measure its performance for individual systems. Comparison of performance of XGBoost and the current Solar Monkey analytical model is provided in table 5.1.

Table 5.1: Individual system metrics comparison for Solar Monkey and XGBoost trained on randomly split data

Individual system metrics	Units	Solar Monkey	XGBoost
min RMSE	kWh	0.371	0.398
max RMSE		9.435	4.618
mean RMSE		1.824	1.355
mean MAE		1.303	0.886
mean MAPE	%	43.62	23.02
mean NRMSE	kWh / kWp	0.425	0.313

As expected, due to lack of system size normalization and choice of RMSE as the evaluation metric, results for individual systems have large variance. It can be observed that XGBoost has individual system RMSE oscillating between around 0.4 kWh and 4.6 kWh. The minimal RMSE of XGBoost is similar to the one of the analytical model while the maximal RMSE of XGBoost is around two times lower. Also, the mean RMSE decreased by around one third. The usage of ML model caused a drop in relative error (mean MAPE) from around 44 % to 23 % which is almost twofold improvement. It is important to notice that measuring performance on

future time steps gives the closest possible test to the real life conditions. Therefore, after model implementation the resulted metrics are very likely to be close to those reported in table 5.1. Distributions of MAPE and RMSE for individual systems can be found in figures 5.1 and 5.2 respectively. It should be noticed that daily results

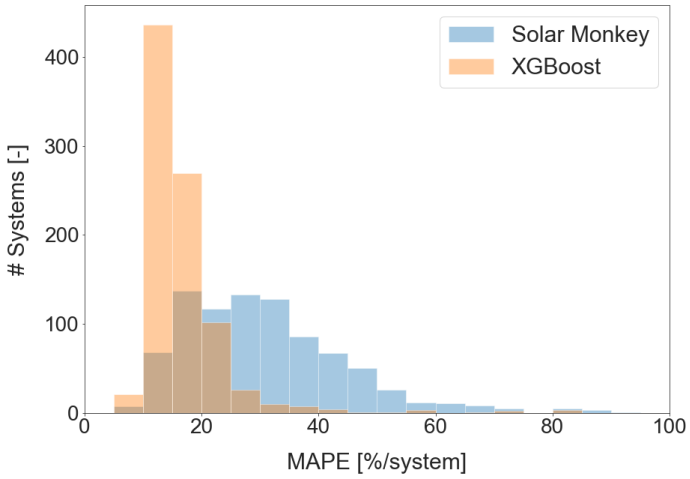


Figure 5.1: MAPE distribution calculated for each system individually

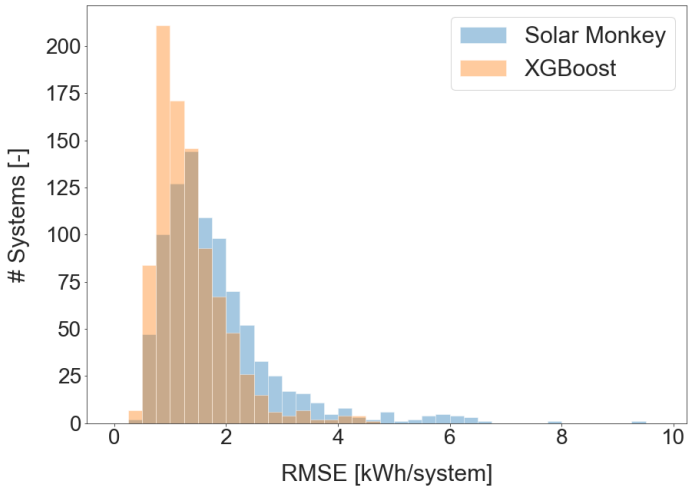


Figure 5.2: RMSE distribution calculated for each system individually

reported for this XGBoost model in table 4.6 were not confirmed and that RMSE and MAE calculated for all predictions are 1.484 kWh and 0.877 kWh respectively. This leads to a surprising observation that hourly predictions aggregated to daily values are worse than those made by the daily XGBoost model. Most likely, that is caused by the utilization of HMM for daily data cleaning, described in chapter 2.

5.2 GROUP SHUFFLE SPLIT

In this approach all samples corresponding to 650 PV systems were assigned to the training set while all samples corresponding the remaining systems were assigned to the test set. The model was trained on full year of data and time steps between July 1st, 2018 and June 30th, 2019. In this section its evaluation on past and future time steps is described. Past time steps imply the model is tested on previously unseen systems, but corresponding to the same time as the training samples. The analysis of the second case was performed for the same model, but evaluated on

time steps between July 1st, 2019 and May 30th, 2020, hence future with respect to the training set. Therefore, all 1,102 systems were split into seen and unseen and evaluated separately. This way it can be determined whether the model makes better predictions for the systems it has previously seen. It is important, as training on full data base consisting of more than 10,000 PV systems is infeasible using the developed code. XGBoost training on the entire data base requires utilization of parallel computing or incremental learning which are beyond the scope of this project. However, this section provides evidence that pursuing such approach is not necessary, as the created XGBoost model scales well to the previously unseen systems.

5.2.1 Past time steps

A comparison of this model with the current Solar Monkey model can be seen in table 5.2. The mean per-system RMSE is the most important metric and based on its value it can be concluded that the model utilizing XGBoost algorithm is around two times better than the commercially used analytical model. The remaining metrics are better in case of XGBoost as well. Percentage error distribution of predictions

Table 5.2: Comparison of individual system metrics for Solar Monkey and daily aggregated XGBoost model for previously unseen systems and identical time steps in training and testing data

Individual system metrics	Units	Solar Monkey	XGBoost Unseen Systems
min RMSE	kWh	0.809	0.307
max RMSE		10.262	5.737
mean RMSE		2.074	1.025
mean MAE		1.536	0.714
mean MAPE	%	118.9	24.23
mean NRMSE	kWh / kWp	0.469	0.22

made by XGBoost was plotted in figure 5.3. Clearly, it is narrower and has a sharper peak around zero than similar distribution corresponding to the analytical model. Comparison of individual RMSE and MAPE for each system can be seen in figures

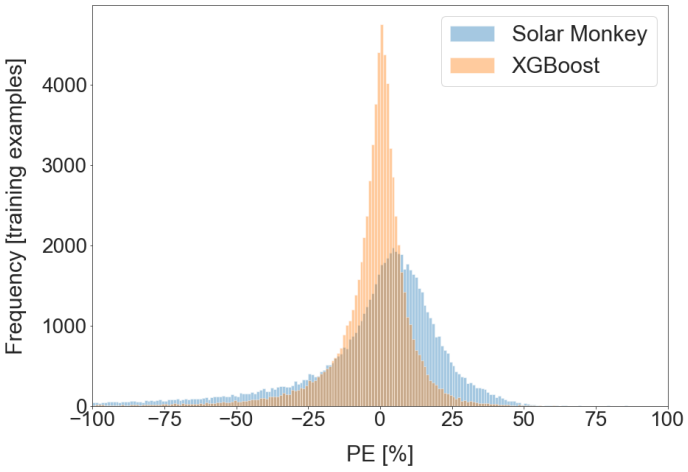


Figure 5.3: Comparison of percentage error distributions

5.4 and 5.5 respectively. It can be noticed that even for previously unseen systems XGBoost outperforms the current analytical model. The distributions of individual

RMSE and MAPE are shifted towards the left with respect to the Solar Monkey model. This implies less severe outliers and smaller relative errors.

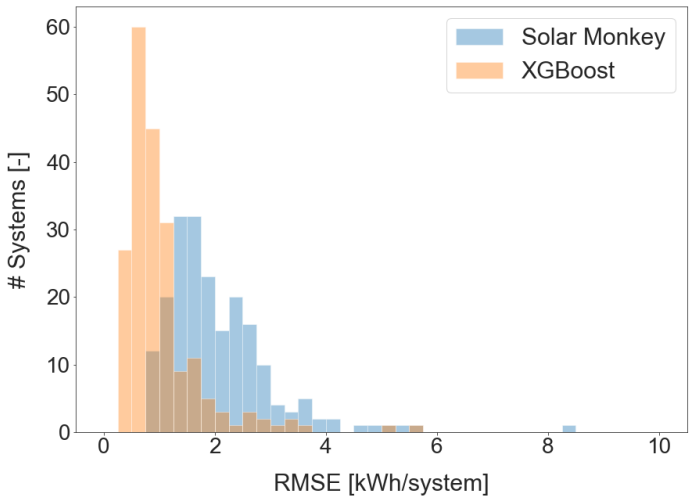


Figure 5.4: RMSE distribution calculated for each system individually

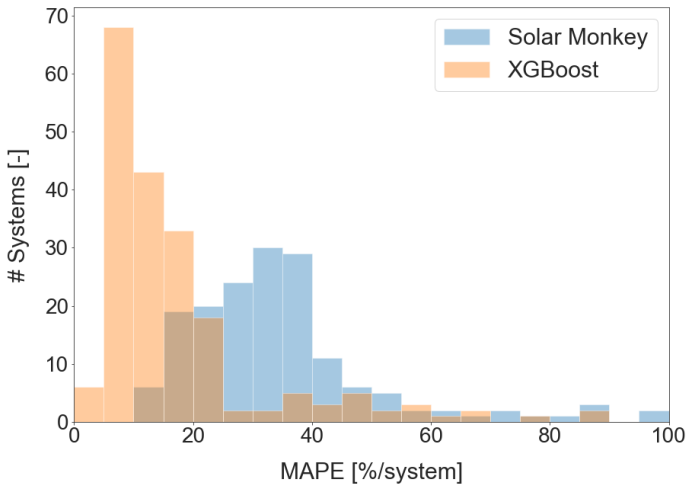


Figure 5.5: MAPE distribution calculated for each system individually

It was assumed that using the same time steps for training and evaluation allows the model to learn and apply the pattern corresponding to a particular time period. The analysis of future time steps described next, in theory, should provide worse results, as the model has never seen the test time steps before. However, the model described in the following subsection should be better than the one described in section 5.1, as it is trained on full year of data and therefore should be able to learn the seasonal pattern.

5.2.2 Future Time steps

Using 650 systems for training and all available systems for evaluation allowed the comparison of XGBoost performance on previously seen and unseen systems. The model created using *GroupShuffleSplit* was evaluated on future time steps and the results of this evaluation together with comparison with the analytical model can be seen in tables 5.3 and 5.4. Once these tables are compared it can be seen that there is almost no difference between the metrics for seen and unseen systems. XGBoost does not learn individual system properties, as type of modules, their decay per year

and other system parameters were rejected during feature selection. The model managed to learn a general pattern and use it to make predictions for system it has never seen before. However, it can be seen that the model trained using *GroupShuf-*

Table 5.3: Comparison of individual system metrics for the Solar Monkey model and XGBoost trained using *GroupShuffleSplit* and evaluated on future time steps and **seen** systems

Individual system metrics	Units	Solar Monkey	XGBoost
min RMSE	kWh	0.506	0.502
max RMSE		6.494	6.098
mean RMSE		1.83	1.656
mean MAE		1.316	1.203
mean MAPE	%	39.26	35.43
mean NRMSE	kWh / kWp	0.432	0.382

Table 5.4: Comparison of individual system metrics for the Solar Monkey model and XGBoost trained using *GroupShuffleSplit* and evaluated on future time steps and **unseen** systems

Individual system metrics	Units	Solar Monkey	XGBoost
min RMSE	kWh	0.563	0.644
max RMSE		9.435	5.449
mean RMSE		1.818	1.666
mean MAE		1.275	1.204
mean MAPE	%	43.84	39.04
mean NRMSE	kWh / kWp	0.408	0.377

fleSplit did not perform well on future time steps. It still outperforms the analytical model in all metrics except the minimal RMSE, but its results are significantly worse than those presented in table 5.2. The decrease in performance was expected, but its magnitude is much higher than assumed. Clearly, the presented results are not realistic and do not match with the previous findings. A possible explanation of this disappointing outcome is the systems selected for training were of different sizes than the systems in the test set. Therefore, the distribution of training data did not resemble the distribution of the test data. For these reasons it is assumed that the results presented in this section are significantly lower than would be if the training systems were more carefully selected. Therefore, the distributions of MAPE and RMSE are not presented. Due to time constraints this issue was not investigated further and is left for future analysis. More attention should be given to selection of the training systems and it should be noticed that the comparison of seen and unseen systems is inherently biased, as its outcomes are system size dependent.

5.3 WEATHER ANALYSIS

In order to capture seasonality, for weather analysis the model trained using all year of data was utilized (vide section 5.2.1). It was investigated whether XGBoost has error dependent on the weather parameters. In case of analytical models errors are often associated with particular time of year or ambient conditions. It was initially assumed that it is not the case for machine learning models which are based on statistics. Graphs depicting Absolute Percentage Error (APE) vs. GHI, cloud cover-

age, wind speed and ambient temperature can be found in figure 5.6. Next to the

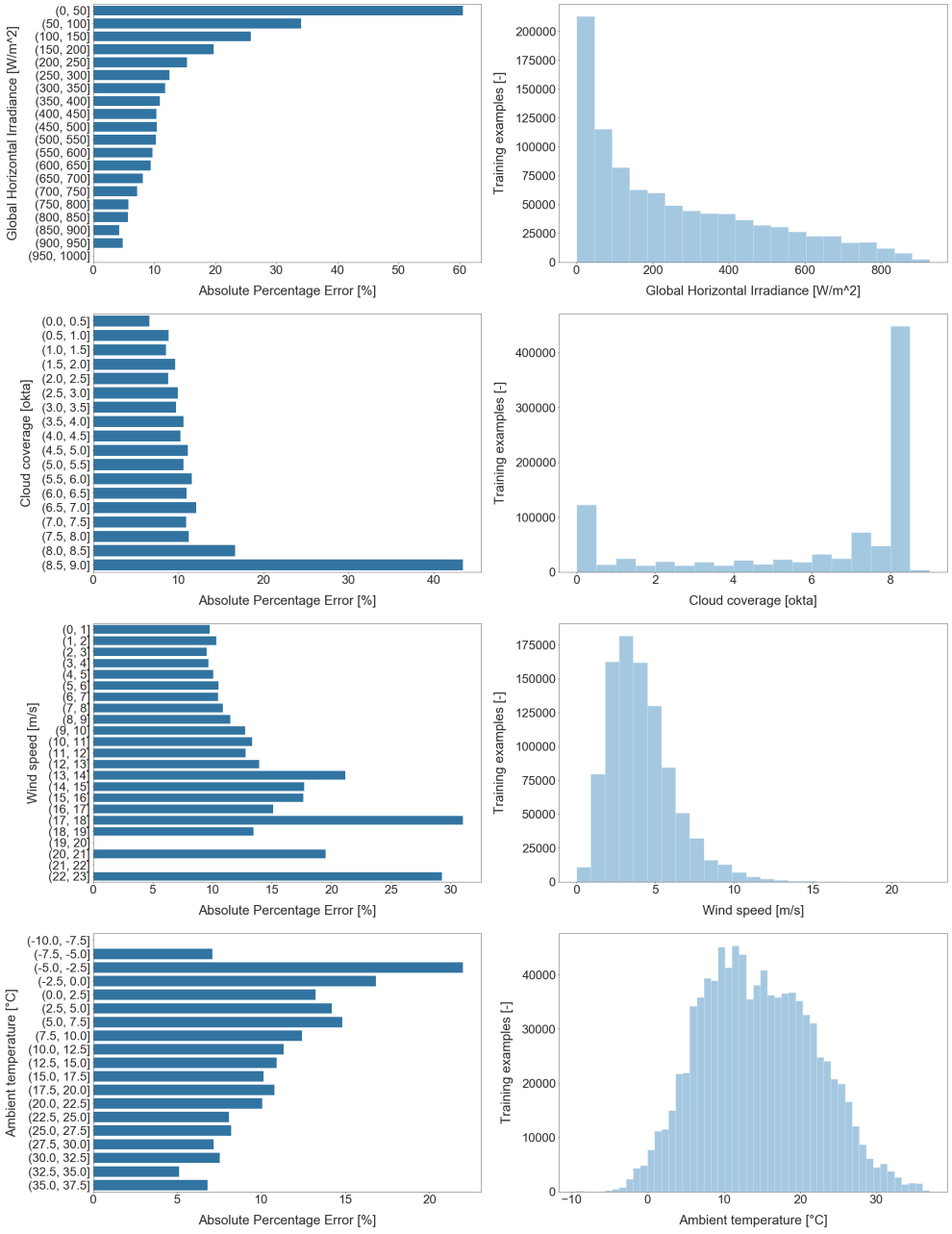


Figure 5.6: Comparison of different weather factors with respect to residual error

histograms of error, sample distributions were plotted to indicate the importance for quality of predictions. For example, if in a certain range a weather parameter occurs more frequently, high APE in that range would be particularly worrying. This is the case for GHI values below 50 W/m^2 for which APE reaches 60 % (vide 5.6). Cloud coverage, wind speed and ambient temperature have the largest values of APE which are not associated with the most frequent values. In case of cloud coverage large APE corresponds to values above 8.5 okta which are very rare. It can be seen that each range of cloud coverage below eight okta has similar APE, but the relative error rapidly increases above that threshold. This implies the XGBoost model performs worse in case of completely overcast sky than could be concluded from table 5.5. Large APE exceeding 25 % occurs for wind speeds between 22 m/s and 23 m/s and between 17 m/s and 18 m/s which are relatively rare. Ambient temperature has the highest APE between -5 °C and -2.5 °C which also occur very rarely. Additionally, both ambient temperature and wind speed have very low fea-

ture importance which can be seen in figure 4.1. It should be kept in mind that solar yield is highly non-linear and links between APE and parameters in figure 5.6 do not provide full overview. It is true that APE has the largest values for certain negative ambient temperatures, but does not mean it is caused by them. For example low ambient temperature can be correlated with the small values of GHI which in turn is of high importance for the model. Keeping in mind that GHI is the single most influential feature for XGBoost based model, the alignment of its large APE and occurrence histogram is the most worrying.

Next, in order to confirm that predicting solar yield under cloudy conditions is much more difficult than under clear sky, an analysis similar to the one conducted by [Durrani et al., 2018] was performed. Its results can be seen in table 5.5 where the performance of the XGBoost and the persistence model was compared under clear, partially cloudy and completely overcast sky. It can be seen that the persistence model performance is irrespective of cloud coverage and its MAE, RMSE and MAPE remain similar for all analyzed conditions. R^2 worsens by around 0.1 which implies insignificant decrease in the quality of predictions. These results are not surprising, as the persistence model simply assumes that yield will persist for another hour and is not influenced by cloud coverage. However, XGBoost clearly provides better results under clear sky than for partially cloudy or completely overcast sky. Its MAE increased from 0.149 for clear sky to 0.236 for partly cloudy sky. Similar is the case for RMSE and MAPE which also worsened by around half. R^2 dropped from 0.95 under clear sky conditions to 0.87 under completely overcast sky which further confirms that XGBoost performance worsens with increasing cloud coverage.

Table 5.5: Comparison of performance for XGBoost and persistence model - hourly XGBoost model with unseen systems in the test set

Nowcasting model	Weather condition											
	Clear sky (<1 okta)				Partly cloudy (1 - 7 okta)				Cloudy day (>7 okta)			
	MAE	RMSE	MAPE	R^2	MAE	RMSE	MAPE	R^2	MAE	RMSE	MAPE	R^2
XGBoost	0.149	0.281	6.44	0.95	0.236	0.377	10.73	0.9	0.233	0.374	11.88	0.87
Persistence Model	0.553	0.73	27.34	0.69	0.577	0.768	28.64	0.6	0.567	0.76	31.73	0.46

Next to the hourly model, weather analysis for the hourly results aggregated to daily values was performed in order to provide valid comparison with the current Solar Monkey software. Results of such comparison can be seen in table 5.6 where the clear trend described previously is no longer present. RMSE and MAE for samples corresponding to completely overcast sky are significantly lower than for clear sky conditions. The only metric constantly worsening with increasing cloud coverage is MAPE. The reason for the lack of visible correlation in absolute metrics is the daily aggregation of data, as for obtaining the results in table 5.6 daily medians of cloud coverage were used. From tables and distributions of residuals presented in

Table 5.6: Comparison of performance of XGBoost and Solar Monkey model - configuration with unseen systems in the test set

Nowcasting model	Weather condition											
	Clear sky (<1 okta)				Partly cloudy (1 - 7 okta)				Cloudy day (>7 okta)			
	MAE	RMSE	MAPE	R^2	MAE	RMSE	MAPE	R^2	MAE	RMSE	MAPE	R^2
XGBoost daily aggregated	0.733	1.267	6.04	0.99	0.956	1.544	6.93	0.98	0.613	1.136	33.99	0.98
Solar Monkey	2.15	3.076	15.02	0.94	2.009	2.825	13.88	0.93	1.26	2.016	177.42	0.95

this section it can be concluded that XGBoost performance is independent of wind speed and ambient temperature, but depends heavily on GHI and to some extent on cloud coverage. Based on the results of the hourly metrics, the analyzed model performs around 50 % better in case of clear sky conditions than under partly overcast

sky. Results presented in table 5.5 confirm that the utilized cloud coverage data is insufficient to precisely capture real life changes.

5.4 ERROR ANALYSIS

Similarly to the weather analysis, error analysis was performed for the model described in section 5.2.1 which includes seasonality. It was already described that utilization of the squared error loss function favours large yields, that is large PV systems and sunny hours. To further verify this hypothesis, observed yields vs. their corresponding APE were plotted in figure 5.7. It can be clearly seen that APE

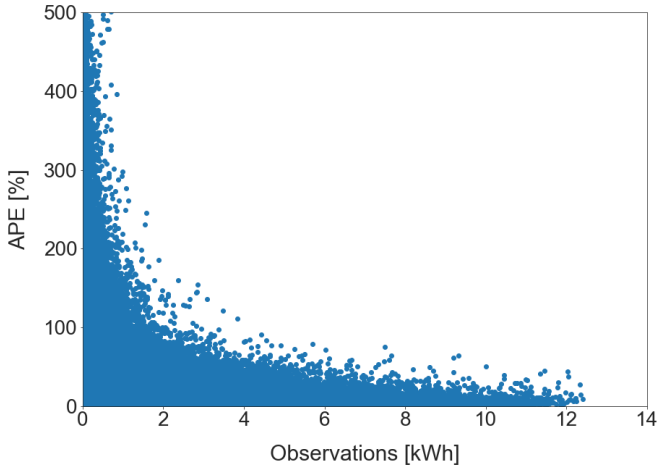


Figure 5.7: Observed yield vs. absolute percentage error

has values far exceeding 100 % for yield values below 2 kWh which confirms the initial assumption. In order to investigate it further APE with respect to hour of a day and month was plotted in figures 5.8 and 5.9 respectively. It can be seen

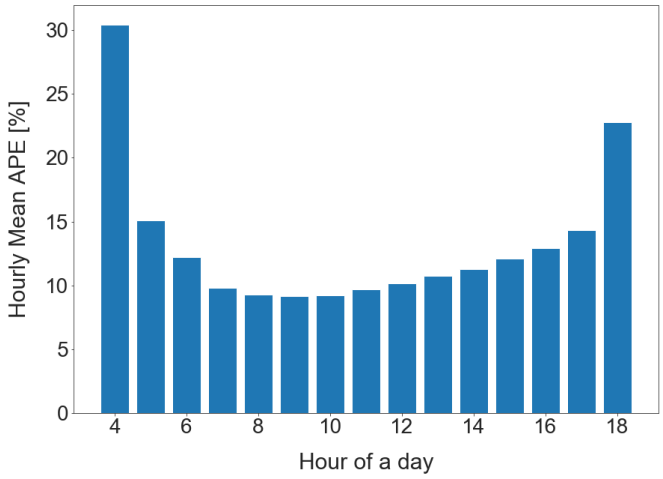


Figure 5.8: Error analysis with respect to time of the day

that the model has the largest relative error for December followed by January. The reason for that might be these two months have the least irradiance in the whole year and therefore correspond to the smallest yields. Analysis of APE with respect to hours has shown that the relative error is the largest for hours just after sunrise and just before sunset. Final consequence of using squared error loss function can be seen in figure 5.10 where residuals follow clear, quadratic increase with respect to system size.

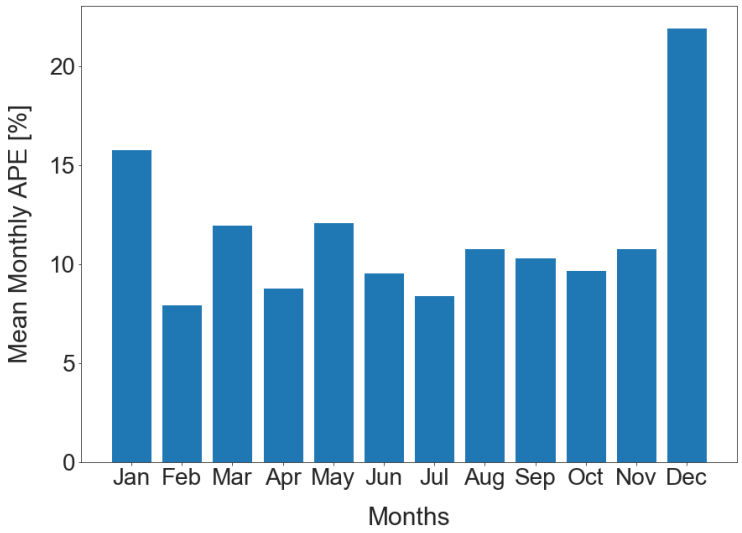


Figure 5.9: Error analysis with respect to months

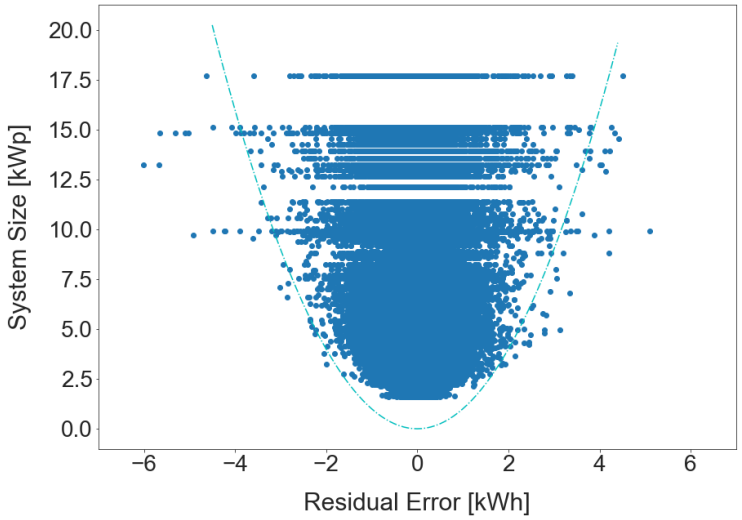


Figure 5.10: Residuals vs. system size

5.5 WEIGHT NORMALIZATION

In order to prevent the XGBoost model from favoring large systems, normalization was used. In figures 5.11 and 5.12 it can be seen that PV systems of smaller size have higher relative errors while larger systems have higher absolute errors. The cause of this problems is the loss function utilized for models training, *squared error*, being system size dependent. Large systems naturally have large errors which are additionally squared making the algorithm focus on them. This causes smaller systems to have higher relative (percentage) errors. This issue was attempted to solve by assigning weights to all training examples. The weights were inversely proportional to system size following the equation: $weight = \frac{1}{system\_size}$. Next, training evaluation metric was changed to MAPE which ensured that relative, rather than absolute error is minimized. E.g. 5 % error over 5 kWh error for all systems is preferred, as absolute error of 5 kWh for small systems would significantly harm the quality of their predictions. Change in results due to introduction of weights and MAPE as boosting evaluation metric can be seen in figure 5.13.

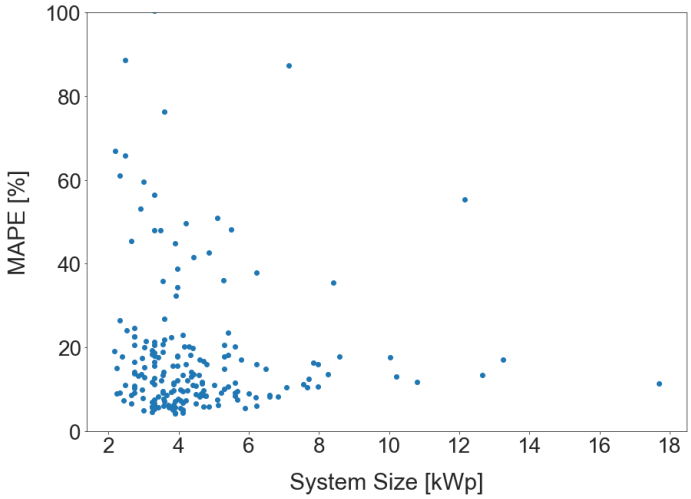


Figure 5.11: System size vs. MAPE

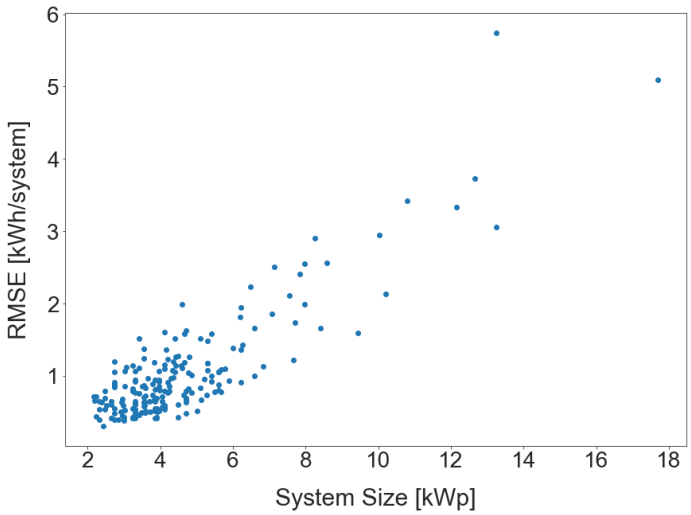


Figure 5.12: System size vs. RMSE

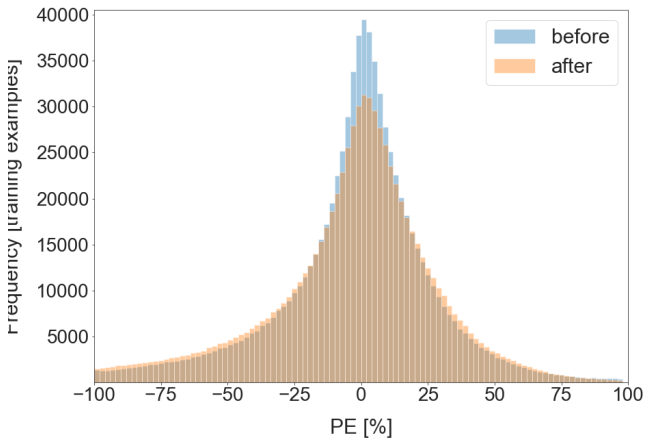


Figure 5.13: Percentage error distributions before and after adding MAPE and weight-based normalization to the XGBoost training process

After using weights to correct training, percentage error distribution has flattened having more large errors. It was observed that XGBoost used 1,000 boosting rounds to optimize RMSE, but only 23 rounds to optimize MAPE, as *early stopping* was triggered. This means MAPE did not improve for 5 rounds and the model fitting was interrupted. Clearly, assigning weights did not solve the problem of large relative errors for small systems. Even though it was known from the beginning that using weights is theoretically incorrect, as all predictions are equally important, this approach was expected to provide improved practical results. Surprisingly, even after removing weight normalization and using MAPE as stopping metric, the results were worse than those obtained for the initial model using the squared error loss. Therefore, question regarding system size normalization remains open.

One might wonder why MAPE was not used from the beginning and the initial focus was on RMSE. It might also seem unclear why a different loss function, utilizing system size, was not pursued. Firstly, MAPE is sensitive to outliers and can have very large values for small absolute errors. E.g. 0.5 kWh prediction for 0.1 kWh observation corresponds to 400% error while impact of such event on overall model performance is negligible. *sklearn* library allows to use two evaluation metrics [Pedregosa et al., 2011], but only for tuning purposes, not for training. Secondly, MAPE cannot be used as loss function which has to comply with strict mathematical requirements, e.g. it has to be twice differentiable. After researching available loss functions [Grover, 2018] it was concluded that none of the common solutions is suitable for the task of large scale solar yield monitoring. Several attempts to develop custom loss function for this project were made, but all of them failed. For these reasons regular *squared error* loss was utilized.

In this chapter two separate models utilizing XGBoost algorithm and developed for hourly data containing GHI were described. First one was created on randomly picked 80 % of the training set and the other utilized full year of data corresponding to 650 PV systems. Both of them were evaluated on future time steps, but only the first one provided promising results which are on average two times better than those of the analytical model. Failure of the *GroupShuffleSplit* for future time steps is assigned to mismatch between training and testing data distributions. Nevertheless, its evaluation on past time steps revealed that XGBoost algorithm is scalable to unseen systems and therefore developing its paralellized version is not necessary. Performing analysis on data with future time steps made it possible to assess individual systems predictions. It was discovered that models based on XGBoost algorithm are mistaken the most for small observed yields and therefore for small irradiance and large cloud coverage values. There is also clear, negative correlation between relative error and system size caused by using squared error as the loss function. Unfortunately, the presented normalization method failed to achieve similar quality of predictions for all systems. Despite careful investigation of several loss functions none of them was identified as suitable for the task of PV yield nowcasting for multiple systems. All attempts of developing a custom loss function combining absolute and relative error have failed.

6 CONCLUSIONS AND DISCUSSION

The principal objective of this study was to provide high quality photovoltaic yield predictions for the purpose of system monitoring. Predictions are made for the presence, not for the future, hence it is the task of solar yield nowcasting. Five different methods ranging in complexity were presented and compared with the commercially available software. Among them, industrial implementation of Random Forest and XGBoost models is justified, as they both outperform current Solar Monkey software. However, in order to prevent large errors as much as possible, XGBoost is preferred. Additional argument supporting this choice is short training time. Two versions of XGBoost model were created, one for hourly and one for daily data resolution. Daily model is used only when no hourly data is available. The best hourly model is mistaken by 0.877 kWh per prediction on average, while currently used commercial software is mistaken by around 1.5 kWh per prediction being around two times worse. Despite superior performance, the best developed model suffers from large relative errors for small measured yields. After this brief summary, answers to the research questions are provided below.

RQ1: Which machine learning algorithm provides the best performance for the task of PV yield prediction?

It is the Extreme Gradient Boosting which turned out to be the most suitable algorithm for the task of PV yield nowcasting. It obtained the lowest RMSE in all three analyzed data sets. Moreover, it scales well to previously unseen systems which allows to avoid development of its parallelized version and makes it particularly suitable for commercial applications. It has to be noticed however, that Random Forest has results very close to XGBoost. The reason for that can be similar working principle, as both algorithms utilize decision trees. Random Forest seems to have more frequent high quality predictions, but once significant errors happen they are of higher magnitude. The reason for choosing XGBoost is the desire to avoid such large errors as much as possible. It is better to obtain predictions further from ideal on average if that allows to prevent large errors from happening. Two other developed ML models, ElasticNet and Polynomial Regression, turned out to be worse than the current commercially used software. They have large MAPE which is unacceptable for solar yield nowcasting.

RQ2: Should machine learning methods replace the analytical approach?

As proven in this study, machine learning models have several advantages: superior performance, short prediction time and the ability to improve with experience. At the same time several factors limit their application. ML models have relatively high computational requirements, long development time and are of high complexity which makes them harder to debug. Sometimes it is not possible to understand why a model is mistaken for a particular case. Additionally, upscaling for large data sets is problematic and distribution of the training data limits the application of ML. The latter means that model trained on data from the Netherlands might not perform well in countries with different climate, e.g. Spain. Therefore, it is difficult to apply ML models in new markets where no large data base for multiple systems is available.

Analytical models are easy to debug, immediately applicable in new markets, do not require large data set and in theory should provide nearly ideal predictions for per-second data resolution. Their drawbacks are longer time required to make predictions, lower prediction quality and inability to learn. Even though in this report it is often stressed that ML models outperform the analytical model, the latter will continue to play an important role in Solar Monkey operations. It is indispensable in new geographical locations where large, clean data base is not available. Also, the analytical predictions can be compared with XGBoost predictions to identify PV systems with incorrect system parameters such as string configuration, number of modules etc.

RQ3: What are the common variables chosen by all models and what is the best set of inputs?

Values of previous yields together with irradiance and total installed power have dominated feature importance graphs with cloud coverage following them. Solar position including azimuth, altitude and zenith and day of year have much lower relevance. It is assumed that when using data for longer time span than a year, day of year feature would be assigned larger weight, as it would incorporate seasonality. Type of solar modules, system age and ambient conditions such as wind speed, precipitation and ambient temperature were assigned surprisingly small weights. System specific parameters such as Nominal Operating Cell Temperature (NOCT), system latitude and longitude have almost negligible importance as well. Also shading, expressed by the obstacle factor and based on horizon maps, had far lower impact than expected. Possible reasons could be averaging module shading values for each system or obsolete 360° maps of surroundings. The optimal set of inputs together with their weights assigned by the best developed model is depicted in table 6.1. Analyzing the impact of time series features and understanding ML working principles it can be concluded that the developed models should not be utilized to verify the existing solar engineering theory. In analytical models PV yield yesterday has no influence on PV yield today. However, for ML models time component is of paramount importance. Apparently, they notice that weather changes usually are not abrupt, but occur gradually. Therefore, they can use historical yield and irradiance values to learn the pattern in solar data. Some ML models, e.g. Ridge regression (vide appendix E), might reduce importance of all but one most influential parameter close to zero. This does not mean however, that other variables have no

Table 6.1: The best model set of features with their corresponding weights

Feature	Importance [%]
GHI	38.8
yield 1d	26.48
yield 2d	10.59
total watt peak	9.55
yield 5d	3.65
yield 3d	3.23
cloud coverage	2.77
yield 4d	1.58
sun altitude	1.18
day of year	0.75
ambient temperature	0.38
sun azimuth	0.35
module orientation	0.33
wind speed	0.19
precipitation	0.15

physical impact on solar yield and is just a result of purely statistical operation.

RQ4: Is developing a single general model better than developing models for all systems individually?

In order to answer this question a comparison between this study and the one by [Visser, 2018] should be made. Even though E. Visser focused on yield forecasting, rather than nowcasting and utilized 15 min data resolution, her study touches upon multiple issues relevant for both projects. Similar conclusions are choice of XGBoost algorithm as the best for predicting photovoltaic yield and discovery of correlation between relative error and system size. However, E. Visser recommended usage of XGBoost models built for each PV system individually. That conclusion is challenged here, as individual models are unable to learn the influence of module and inverter types and other cross-system parameters. Also, training individual models for each out of 10,000 systems is cumbersome and sub-optimal in production environment. Detailed E-metrics for hourly resolution reported in both studies were compared in table 6.2. Results for forecasting and nowcasting models can be compared only because E. Visser discovered that usage of forecasted and historical weather data provides similar results. It can be seen that the general model has worse results

Table 6.2: Performance comparison of one XGBoost model for multiple systems and XGBoost model built for each system individually [Visser, 2018]

	Data Resolution	Forecast Horizon	Model Type	E 10	E 50	E 100	E 500
Nowcasting	1 h	0 h (now)	General	23.55%	54.66%	71.35 %	96.66%
Forecasting	15 min	2 h	Individual	38.6%	55.4%	67.6%	95.6%

for E_{10} and E_{50} metrics, but outperforms the individual model in E_{100} and E_{500}. It should be stressed that the forecasting model utilizes four times better data resolution which gives it significant advantage. Nevertheless, the results are very similar which indicates that developing one model for all systems does not provide significant difference while it saves time and effort. Also, general model can make predictions for new systems immediately without the delay for individual system data gathering. Therefore, it can be concluded that developing a general model is the approach of choice.

Before the last research question is answered the reasoning behind applying domain knowledge for ML inputs preprocessing is provided. The entire idea of machine learning is to let it discover the equation linking various parameters with PV yield on its own. One might wonder whether preprocessing of inputs using domain knowledge contradicts this idea. Feature engineering aims to increase the amount of information available in the data set. GPOA contains more information than GHI, as modules are usually tilted and irradiance on horizontal surface does not correspond to the real irradiance received by solar modules. GPOA calculation was utilized, because it is purely trigonometrical and does not contain much uncertainty. Other parameter which could increase the amount of information in the data set is the instantaneous inverter efficiency provided by the SNL model [King et al., 2007]. It would be an improvement, as models developed in this project are aware only of the maximal inverter efficiency, but do not acknowledge that inverter efficiency changes over time. Domain knowledge can be useful also during data filtering, e.g. for GHI larger than 10 W/m^2 yield is supposed to be zero, as generated power is usually below the inverter startup power. Non-zero yield values corresponding to GHI below 10 W/m^2 should be removed. Similarly, samples with zero yield for high values of irradiance should be removed as well.

In some cases applying solar theory for preprocessing of ML inputs does not seem

justified. Soiling equations described by [Nepal et al., 2019], [Maghami et al., 2016] and [Lindig et al., 2018] depend on empirical, location-dependent soiling coefficient which could harm model performance. Ageing equations described by [Manganiello et al., 2015] and [Quansah and Adaramola, 2018] cannot incorporate individual defects in silicon material and the pace of ageing highly dependent on ambient conditions. The usage of Fluid Dynamic model does not seem justified neither, as it assumes uniform temperature distribution inside a semiconductor material. That is not true due to non-uniform material structure and temperature differences caused by partial shading. All models mentioned in this paragraph have analytical value, but are not a good fit for ML techniques due to the included assumptions.

RQ5: Is GPOA precalculation beneficial for ML models? Would using its raw components provide better results?

Before the project started data set containing GPOA had been assumed to be the most promising. According to [Massaoudi et al., 2019] "relying on domain knowledge contributes significantly to the prediction accuracy". This was not confirmed, as contrary to the initial belief, pre-calculating GPOA did not improve model performance. GPOA was calculated using two configurations - with and without shading. Surprisingly, including shading factor further decreased model performance. It is assumed that disappointing results of GPOA and shading calculations were caused by lack of data cleaning and large data resolution. GPOA calculation is analytical, as features follow strictly formulated mathematical formula with hidden assumption of continuous data. It could work also if average hourly values of continuous data were given. However, that is not the case. For large data resolutions input values were observed at a given moment and the measured parameter could have arbitrarily large fluctuations between the measurements. ML utilizing raw features is not constraint by the analytical GPOA equation and therefore discovers its own empirical formula better fitting the pattern. For this reason ML models utilizing raw components of GPOA perform better than those utilizing GPOA itself. Following this reasoning, the finer the data granularity the more beneficial it is to pre-calculate GPOA. The reason for harmful influence of shading calculations might be obsolete horizon views and calculating average shading for all PV modules. Next to the research questions, several other topics related to the advanced analysis should be covered. The following questions were formulated during research and are related to advanced properties of the XGBoost algorithm and learning principles of ML models.

When should a system be added to the training set? For all ML models training and testing data should come from the same distribution. If small systems dominate in the training set the algorithm might treat large systems as noise. This could result in significantly worse predictions for under-represented systems and is the case also for system age, irradiance and all other features. The larger the feature weight, the more important it is to ensure its large variance in the training set. A system should be added to the training set only if it corresponds to a case which is poorly represented or not represented at all.

Did the model learn individual system parameters? It was discovered that created models scale well to previously unseen systems and that one general model is as good as the individual models for each system. Therefore, it might seem like XGBoost is unable to learn individual system properties. However, that is not the case, as these parameters are already incorporated in the historical yields.

Does the model provide better predictions for systems with larger number of samples in the training set? Feature *system age* already describes number of samples corresponding to a particular system. The older the system is the larger is its

age in days and the more samples it has. In figures 4.1 and 4.2 it can be seen that this feature was not selected for the XGBoost algorithm which implies the quality of predictions is independent of the number of samples corresponding to each system. Therefore, assigning weights to under-represented systems is not required.

Can fewer systems with data corresponding to longer time range be used for model training? In this project data for around 1,000 systems for one year was utilized. Using data for two years and 500 systems is likely to provide similar results, as long as the training set is possibly diverse. Obviously, a threshold exists and using data for one system and 1,000 years would provide disappointing results, as the training distribution would not resemble the distribution of the test set.

How many systems and with how long time range are required for the model learning to saturate? XGBoost learning saturates around 40 % of the training set, that is around 1.28 mln samples. In the analyzed data set each system has 4,609 samples corresponding to non-zero irradiance in a year. Dividing the total number of required samples by samples per system it can be concluded that data for around 278 PV systems for an entire year is required for the XGBoost algorithm to reach the quality of predictions reported in this thesis.

Finally, it is concluded that the best performing model developed in this project was described in section 5.1 and can provide approximately two times better predictions than the current commercially available software. However, despite promising results the model can be further improved.

7 RECOMMENDATIONS

In this chapter recommendations both for academic research and further development of the commercial monitoring service are provided. Improvements in the method, data quality and best practices are described.

7.1 SOLAR MONKEY

This section focuses on recommendations for the company. Changes in the input data, hints on preserving the developed code and remarks on deep learning and anomaly detection are provided.

5 min resolution Power production by solar modules is a phenomenon dependent on multiple rapidly changing parameters. Hourly data resolution is insufficient to precisely model cloud coverage and module temperature. Cloud coverage changes are much more frequent than once per hour, as they are highly dependent on wind speed which is significant in the analyzed region (the Netherlands and Belgium). As presented in chapter 5, the created model performs worse for overcast sky than for clear sky which implies cloud data is of low quality. Taking into account the trade-off between hardware and physical requirements, 5 minute data resolution is recommended. According to [Smets et al., 2016] PV module thermal inertia is 7 minutes, hence 5 min data resolution should be sufficient to accurately model it.

Solar engineering for ML inputs pre-processing As a result of the drawn conclusions, GPOA calculation should be repeated for finer data resolution. Also, for data granularity in the order of minutes inverter efficiencies calculated using SNL model [King et al., 2007] should be utilized. Blending analytical assumptions with ML techniques is a recommended path for further research. For fine data resolution inverter clipping implementation is worth investigating. Also, domain knowledge should be used both for data filtering and correction of predictions. For example the latter could be limited to the inverter maximal power.

Better cloud data It was realised that KNMI cloud coverage data utilized in this project imply that the sky is completely overcast for vast majority of time. That seems unrealistic and therefore different cloud coverage data source, such as Meteonorm, could be utilized. It is recommended to investigate the usage of sky images or *peer-to-peer* cloud coverage information based on surrounding systems yields. Solar Monkey monitors multiple systems in the geographical region of interest which makes utilization of peer-to-peer techniques possible. New cloud coverage data do not have to be numerical and can consist of sky images which can be processed by CNN or other deep learning techniques.

Data quality issues Due to the broad scope of this project it was impossible to address all issues in detail. Especially, hourly data filtering could be improved. Three data configurations: without timeseries features, with timeseries features and with timeseries features and data filtering, were tested. Rough analysis revealed that the usage of data filters worsened performance of the ML models. More attention should be given to this issue in the future, as **it is the quality of inputs rather than**

hyperparameter tuning which influences models performance the most. No sensitivity analysis of the data filters was performed and the optimal choice was identified by trial and error, without cross-validation. Sensitivity analysis should be done for each type of corruption detection separately. Data cleaning is of paramount importance, as no model can provide high quality predictions using incorrect inputs. It is non-trivial, time-consuming issue often requiring usage of complex statistics. Therefore, it is recommended to prevent data corruption from happening. Systems often delivering corrupt yields should be identified using the HMM, filters described in chapter 2 or through comparison of predictions from the analytical and ML models. System owners should be reminded about guarantee loss due to the poor wi-fi connection, as large, clean database is a foundation for development of new products such as PV yield forecasts. Although not informing about connection malfunctions provides short-term benefits (prevents guarantee payments), it should be avoided to benefit in long-term, by saving time and money on data cleaning and building high-performing AI models. Using clean data provides additional advantage of releasing the burden from employees responsible for system monitoring and resolving issues, thus making their work easier. A system of incentives and penalties should be provided for the installers to motivate them to provide reliable string configurations.

Preserving project code Code developed in this project should be preserved including file paths and versions of used libraries. This would help to resolve doubts and facilitate quick model testing by using pre-made functions from the created library. BRL model [Ridley et al., 2010] should be added to the library in the future, as several different versions of it exist within the company and some of them provide different results for the same inputs.

New markets According to [Lorenz et al., 2009] "solar forecast accuracy depends on the region of evaluation: for instance, RMSE ranged from about 20% to 35% in Spain, reaching 40% to 60% in Central Europe". This implies that some climates are easier to predict than others. Performance of each PV technology with respect to different climates was analyzed by [Quansah et al., 2017]. Models developed in this project for optimal performance have to be trained on data from the same geographical region as data used for making predictions. In other words, both training and testing data should come from the same distribution. When entering new markets it might not be possible to immediately utilize AI approach which requires large, clean data set to make high quality predictions. In such cases the analytical model used by the company so far should be utilized. When using the AI models in large countries with diverse climates, it is reasonable to split data into regions with similar climate and to develop models for each of them separately. Similarly, when entering a country similar to the one where Solar Monkey already operates one of the existing models could be utilized. It was discovered that gathering data for 278 PV systems for one year is sufficient to reach the optimal quality.

Code implementation Both hourly and daily XGBoost models outperform the commercially available software and should be implemented. For the daily data existing HMM should be utilized and developing its new version for the hourly data is recommended. In both cases HMM should be adapted to use predictions made by the XGBoost as inputs. Implementation of these models is important, as the predictions in the production environment should be evaluated on clean data only. Next, XGBoost files developed in this project can be directly loaded and data pipeline providing continuous inputs should be developed. Created code library might facilitate implementation.

Forecasting This work laid foundations for PV yield forecasting product development in Solar Monkey. Work done by [Antonanzas et al., 2016] might be a good

starting point for further research in this area. Despite data availability and company experience, developing a forecasting algorithm would require substantial effort. Economic aspects of forecasting and grid operator requirements should be studied in detail **before** a model is built. Forecasting often utilizes deep learning techniques of complexity higher than machine learning and taking more time to develop. Also, Solar Monkey has little experience in developing neural networks, hence additional time for the initial mistakes is required. Forecasting model should be created on a customer request and customized to his/her needs. **Developing a general forecasting tool might result in difficulties with its commercialization**. It is recommended to obtain a list of desired conditions a forecasting model must meet before development is started. That includes the desired forecast horizon, available inputs and their resolution. Still several questions arise:

- Who would be willing to pay for the PV yield forecasts?
- What is the market value of PV yield forecasts?
- Is Solar Monkey able to challenge the status quo and make forecasting useful for the residential PV system owners?
- Will the developed forecasts be good enough to safely make transactions on the balancing market?

Forecasting might bring a new stream of revenue, but is associated with high production and marketing costs. Selling a new product would require new channels and most likely utility-scale customers. Moreover, transactions at the balancing market are risky, as they involve large cashflows. To answer all the described questions it is recommended to address them in a separate project.

Retraining In order for ML models to learn, they have to be frequently retrained on new data. The ability to learn, that is to improve performance with experience, is a major advantage of ML over other techniques. However, due to learning saturation described previously, potential for improvement of the created models is very limited. Initially it is reasonable to retrain the model once per week, mainly to make sure it incorporates system ageing and changing pattern in solar radiation. There is no need to perform feature selection after research process is finished. Model might require repeating hyperparameter tuning if the data set increases or changes significantly, but that is unlikely on weekly basis. Performing monthly verification seems reasonable. Finally, XGBoost training is very quick and should not take longer than 10 minutes.

7.2 FURTHER RESEARCH

In this section recommendations for further academic research are given. They are related to utilization of a different loss function, performing sensitivity analysis for created data filters and alternative approaches towards anomaly detection among others.

Metrics It is of paramount importance to develop a loss function able to minimize both absolute and relative errors at the same time. Created loss function should be compatible with *xgboost* native library. Model training using NRMSE as evaluation function or using both MAPE and RMSE for model training are also worth further investigation. It is also highly recommended to validate the best model created in this project against *smart persistence* which is a benchmark for the task of PV yield prediction. This would allow to better position the presented findings in literature. Currently, it is difficult to estimate whether the created model is better or worse than the models developed by other researchers.

Open source data set It is worth considering to publish a data set containing weather, system and yield information for 2 years and 50 systems to facilitate research and allow direct results comparison. Significant effort to create standards of PV yield forecasting and nowcasting is put worldwide, but little progress is done. Partnership between Solar Monkey and Delft University of Technology could combine real-life challenge with academic institution credibility and recognition, drawing attention of large group of researchers. A prize for the best performing model could be established and the participants should be obliged to publish their code. This way TU Delft would contribute to solving the issue of solar yield monitoring and Solar Monkey would obtain multiple models at low cost. Publishing data for 50 systems only would prevent direct utilization of the findings by other companies. Obviously, confidentiality issues and ensuring that market competitors would not benefit from the project should be addressed.

Sensitivity analysis and error estimations Detailed sensitivity analysis of different parts of the monitoring service should be performed. What is the gain in final metric due to data cleaning? What improvement can be obtained when using better weather data? There are numerous sources of uncertainty: averaging of weather data and weighting distance between sensors and PV systems, data cleaning for daily and hourly data sets and shading calculations. Estimating sources of error is an important component of decision making process regarding the next programming steps. It facilitates optimal allocation of resources and is essential both for ML and analytical models.

Deep learning This project did not manage to fully utilize the entire data set potential, as all presented models stop improving their performance by the time they see half of the training set. In order to tackle this issue, next developed algorithms should focus on deep learning and neural networks. Linear regressors should be completely skipped, as the persistence model already provides benchmark and they have zero chance to learn non-linear pattern in the data. LSTM neural networks seem to be the most suitable algorithm for this task and developing them is a rational next step. A few simple LSTMs were developed in this project and the associated findings can be found in appendix E. Future attempts should be made using exactly the same data set to directly compare the results, adjusting it only where necessary. Also a CNN could be built if there is need to process sky images and incorporate cloud coverage data.

Anomaly detection This project focused on developing regression models to ensure easy error quantifying. However, the ultimate goal of the monitoring service is anomaly detection which can utilize classification, probabilistic regression or yield-based statistical methods. Classification labels each yield as corresponding to normally operating system (one), or malfunctioning system (zero). Probabilistic regression provides probability that yield is within certain range. Dealing with probabilities is particularly appealing in monitoring, as it provides level of confidence regarding the detected anomalies. These techniques, together with statistical methods, are worth investigating in the future. It is important to notice that Solar Monkey is not informed about the system breakdowns or maintenance, therefore no labels for these events exist in the database. This, together with lack of precise cloud coverage data, significantly limits the development of high quality anomaly detection service.

Timeseries analysis During analysis of feature importance graphs it was discovered that the historical yields were assigned the highest weights. It is a strong indicator that time component has high importance in solar yield nowcasting and forecasting. Therefore, the historical values of yield and irradiance for more than

five past days should be added to verify their influence on the quality of predictions. Initially, values for 14 days could be added and the feature importance graph could be plotted. Usually the further the features are from the presence, the lower importance they have. At certain moment their weights would decrease close to zero and this way the optimal horizon threshold could be determined. Also, high importance of time series features is a strong incentive to focus on methods such as ARIMA and LSTM neural networks.

Parallel computing / incremental learning Based on the metrics calculation for unseen systems, it is not recommended to develop a parallelized version of the XGBoost model in the nearest future. Developing such software in production environment is time consuming and associated with large financial cost. It was not confirmed that XGBoost provides better predictions for the systems it has already seen. Performed analysis for seen and unseen systems revealed similar scores for both data sets. However, direct comparison is difficult in this case, as the outcomes are system size dependent and for this reason it should not be treated as final. Parallelized model requires very similar data pipeline as the regular one and therefore, after implementation, the quality of predictions for seen and unseen systems should be re-evaluated to confirm the findings. It is strongly believed that even if a difference exists it is small and the gain in model performance does not justify significant development effort put into parallelized version of the XGBoost model.

Final model Even though the best model created in this project outperforms the commercial software by a factor of two, it should not be treated as final. It should be clear by now that preparation of inputs is crucial for the task of solar yield nowcasting using ML. Final model should be trained on clean data corresponding to a variety of systems diverse in size, age and weather conditions. The most important features which should be focused on, are irradiance and system size. High variance of these should be ensured to obtain a robust model. Samples in the training set should correspond to the entire year with possibly even number of samples for each month. Final model should utilize also a custom loss function able to address both absolute and relative errors.

In this chapter recommendations for the industry and academia were presented. Future work in Solar Monkey should focus on obtaining higher granularity data, better cloud coverage information and addressing data quality issues. It was concluded that in further horizon developing forecasting service is possible, but will require significant resources. Recommendations for academia focus on using solar engineering theory to preprocess inputs for ML models, sensitivity analysis and most importantly, on developing custom loss function which combines relative and absolute metrics. Deep learning could be studied to develop a better understanding of solar yield forecasting. Investigation of anomaly detection techniques other than regression is also worth pursuing.

BIBLIOGRAPHY

Abdel-Nasser, M. and Karar, M. (2017). Accurate photovoltaic power forecasting models using deep lstm-rnn. *Neural Computing and Applications*, 31:2727—2740.

Agarwal, R. (2019). The 5 feature selection algorithms every data scientist should know. https://towardsdatascience.com/the-5-feature-selection-algorithms-every-data-scientist-need-to-know-3a6b566efd2, accessed: 18.04.2020.

Al-Messabi, N., Li, Y., El-Amin, I., and Goh, C. (2012). Forecasting of photovoltaic power yield using dynamic neural networks. In *The 2012 International Joint Conference on Neural Networks (IJCNN)*, Brisbane, Australia, 10-15 June 2012. IEEE.

Alzahrani, A., Shamsi, P., Dagli, C., and Ferdowsia, M. (2017). Solar irradiance forecasting using deep neural networks. *Procedia Computer Science*, 114:304–313.

Anagnostos, D., Schmidt, T., Cavadias, S., Soudris, D., Poortmans, J., and Catthoor, F. (2019). A method for detailed, short-term energy yield forecasting of photovoltaic installations. *Renewable Energy*, 130:122–129.

Antonanzas, J., Osorio, N., Escobar, R., Urraca, R., de Pison, F. M., and Antonanzas-Torres, F. (2016). Review of photovoltaic power forecasting. *Solar Energy*, 136:78–111.

Aziz, F., Haq, A. U., Ahmad, S., Mahmoud, Y., Jalal, M., and Ali, U. (2020). A novel convolutional neural network-based approach for fault classification in photovoltaic arrays. *IEEE Access*, 8:41889–41904.

Baharin, K. A., Rahman, H. A., Hassan, M. Y., and Gan, C. K. (2016). Short-term forecasting of solar photovoltaic output power for tropical climate using ground-based measurement data. *Journal of Renewable and Sustainable Energy*, 8.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA.

Bowden, G., Barker, P. R., Shestopal, V., and Twidell, J. (85–98). The weibull distribution function and wind power statistics. *Wind Engineering*, 7.

Branco, P., Gonçalves, F., and Costa, A. C. (2020). Tailored algorithms for anomaly detection in photovoltaic systems. *Energies*, 13.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California, USA, August 2016. New York: Association for Computing Machinery.

Dorpe, S. V. (2018). Preprocessing with sklearn: a complete and comprehensive guide. https://towardsdatascience.com/preprocessing-with-sklearn-a-complete-and-comprehensive-guide-670cb98fcfb9, accessed: 18.04.2020.

Durrani, S. P., Balluff, S., Wurzer, L., and Krauter, S. (2018). Photovoltaic yield prediction using an irradiuance forecast model based on multiple neural networks. *Journal of Modern Power Systems and Clean Energy*, 6:255–267.

Ejgar, M. and Momin, B. (2017). Solar plant monitoring system: A review. In *International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, 18-19 July 2017. IEEE.

Ejgar, M., Momin, B., and Ganu, T. (2016). Intelligent monitoring and maintenance of solar plants using real-time data analysis. *Solar Energy*, 125:77–85.

Elsinga, B. and van Sark, W. G. (2017). Short-term peer-to-peer solar forecasting in a network of photovoltaic systems. *Applied Energy*, 206:1464–1483.

Esposito, D. and Esposito, F. (2020). *Introducing Machine Learning*. Pearson Education, Inc.

Fraunhofer Institute for Solar Energy Systems (2020). Photovoltaics report. https://www.ise.fraunhofer.de/content/dam/ise/de/documents/publications/studies/Photovoltaics-Report.pdf, accessed: 18.06.2020.

Gandoman, F. H., Raeisi, F., and Ahmadi, A. (2016). A literature review on estimating of pv-array hourly power under cloudy weather conditions. *Renewable and Sustainable Energy Reviews*, 63:579–592.

Gapminder Foundation (2020). Life expectancy in the netherlands. https://www.gapminder.org/tools/, accessed: 18.04.2020.

Gensler, A., Henze, J., Sick, B., and Raabe, N. (2016). Deep learning for solar power forecasting – an approach using autoencoder and lstm neural networks. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Budapest, Hungary, October 2016. IEEE.

Grover, P. (2017). Gradient boosting from scratch. https://medium.com/mlreview/gradient-boosting-from-scratch-1e317ae4587d, accessed: 1.06.2020.

Grover, P. (2018). 5 regression loss functions all machine learners should know. https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0, accessed: 5.06.2020.

Hale, J. (2019). Scale, standardize, or normalize with scikit-learn - when to use minmaxscaler, robustscaler, standardscaler and normalizer. https://towardsdatascience.com/scale-standardize-or-normalize-with-scikit-learn-6ccc7d176a02, accessed: 18.04.2020.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9:1735—1780.

Ilyas, I. F. and Chu, X. (2019). *Data Cleaning*. ACM Books, New York.

Isaksson, E. and Conde, M. K. (2018). Solar power forecasting with machine learning techniques. Master's thesis, KTH Royal Institute of Technology.

King, D. L., Gonzalez, S., Galbraith, G. M., and Boyson, W. E. (2007). Performance model for grid-connected photovoltaic inverters. *Tech. Rep. SAND2007-5036*.

Kuzmiakova, A., Colas, G., and McKeehan, A. (2017). Short-term memory solar energy forecasting at university of illinois.

Li, J., Cheng, K., Ding, K., and Liu, H. (2020). Feature selection algorithms. http://featureselection.asu.edu/algorithms.php, accessed: 18.04.2020.

Lindig, S., Kaaya, I., Weiß, K.-A., Moser, D., and Topic, M. (2018). Review of statistical and analytical degradation models for photovoltaic modules and systems as well as related improvements. *IEEE Journal Of Photovoltaics*, 8:1307–1316.

Lorenz, E., Remund, J., Müller, S. C., Traunmüller, W., Steinmaurer, G., Pozo, D., Ruiz-Arias, J. A., Fanego, V. L., Ramirez, L., Romeo, M. G., Kurz, C., Pomares, L. M., and Guerrero, C. G. (2009). Benchmarking of different approaches to forecast solar irradiance. In *Proceedings of 24th European Photovoltaic Solar Energy Conference*, Hamburg, Germany, 21-25 September 2009. Munich: WIP-Renewable Energies 2009.

Maghami, M. R., Hizam, H., Gomes, C., Radzi, M. A., Rezadad, M. I., and Hajighorbani, S. (2016). Power loss due to soiling on solar panel: A review. *Renewable and Sustainable Energy Reviews*, 59:1307–1316.

Maitanova, N., Telle, J.-S., Hanke, B., Schmidt, T., Grottke, M., von Maydell, K., and Agert, C. (2020). Machine learning approach to a low-cost day-ahead photovoltaic power prediction based on publicly available weather reports. *Energies*, 13.

Manganiello, P., Balato, M., and Vitelli, M. (2015). A survey on mismatching and aging of pv modules: The closed loop. *IEEE Transactions on Industrial Electronics*, 62:7276–7286.

Massaoudi, M., Chihi, I., Lilia Sidhom, M. T., Refaat, S. S., and Oueslati, F. S. (2019). Pv power forecasting using weighted features for enhanced ensemble method.

Morde, V. and Setty, V. A. (2019). Xgboost algorithm: Long may she reign! https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d, accessed: 19.05.2020.

Nepal, P., Korevaar, M., Ziar, H., Isabella, O., and Zeman, M. (2019). Accurate soiling ratio determination with incident angle modifier for pv modules. *IEEE Journal Of Photovoltaics*, 9:295–301.

Ng, A. (2020). Structuring machine learning projects. https://www.coursera.org/learn/machine-learning-projects?specialization=deep-learning, accessed: 18.04.2020.

Nikolaou, N., Batzelis, E., and Brown, G. (2017). Gradient boosting models for photovoltaic power estimation under partial shading conditions.

Ogliari, E., Niccolai, A., Leva, S., and Zich, R. E. (2018). Computational intelligence techniques applied to the day ahead pv output power forecast: Phann, sno and mixed. *Renewable Energy*, 8:793–800.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pelland, S., Remund, J., Kleissl, J., Oozeki, T., and Brabandere, K. D. (2013). Photovoltaic and solar forecasting: State of the art. *Photovoltaics Power Systems Programme*.

Phi, M. (2018). Illustrated guide to lstms and grus: A step by step explanation. https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21, accessed: 4.06.2020.

Quansah, D. A. and Adaramola, M. S. (2018). Ageing and degradation in solar photovoltaic modules installed in northern ghana. *Solar Energy*, 173:834–847.

Quansah, D. A., Adaramola, M. S., Appiah, G. K., and Edwin, I. A. (2017). Performance analysis of different grid-connected solar photovoltaic (pv) system technologies with combined capacity of 20 kw located in humid tropical climate. *International Journal of Hydrogen Energy*, 42:4626–4635.

Raschka, S. (2014). About feature scaling and normalization – and the effect of standardization for machine learning algorithms. https://sebastianraschka.com/Articles/2014_about_feature_scaling.html, accessed: 21.05.2020.

Raza, M. Q., Nadarajah, M., and Ekanayake, C. (2016). A literature review on estimating of pv-array hourly power under cloudy weather conditions. *Renewable and Sustainable Energy Reviews*, 63:579–592.

Ridley, B., Boland, J., and Lauret, P. (2010). Modelling of diffuse solar fraction with multiple predictors. *Renewable Energy*, 35:478–483.

Rivai, A., Rahim, N. A., Elias, M. F. M., and Jamaludin, J. (2020). Analysis of photovoltaic string failure and health monitoring with module fault identification. *Energies*, 13:1–16.

Rocca, J. (2019). Ensemble methods: bagging, boosting and stacking - understanding the key concepts of ensemble learning. https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205, accessed: 30.04.2020.

Rosling, H., Rosling, O., and Ronnlund, A. R. (2018). *Factfullness - Ten Reasons We're Wrong About the World - and Why Things are Better Than You Think*. Flatiron Books, Leyde.

Sanfilippo, A., Martin-Pomares, L., Mohandes, N., Perez-Astudillo, D., and Bachour, D. (2016). An adaptive multi-modeling approach to solar nowcasting. *Solar Energy*, 125:77–85.

Schuler, A. (2019). Ngboost and prediction intervals - what is probabilistic regression and how should you interpret probabilistic predictions? https://towardsdatascience.com/interpreting-the-probabilistic-predictions-from-ngboost-868d6f3770b2, accessed: 30.04.2020.

Sharadga, H., Hajimirza, S., and Balog, R. S. (2020). Time series forecasting of solar power generation for large-scale photovoltaic plants. *Renewable Energy*, 150:797–807.

Sharma, N., Chakrabarti, A., and Balas, V. (2020). An integrated fault classification approach for microgrid system. *Data Management, Analytics and Innovation. Advances in Intelligent Systems and Computing*, 1042:41–56.

Sharp, T. (2020). An introduction to support vector regression (svr) using support vector machines (svms) for regression. https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2, accessed: 28.05.2020.

Shin, T. (2020). An extensive step by step guide to exploratory data analysis. https://towardsdatascience.com/an-extensive-guide-to-exploratory-data-analysis-ddd99a03199e, accessed: 18.04.2020.

Smets, A., Jager, K., Isabella, O., van Swaaij, R., and Zeman, M. (2016). *Solar Energy: The Physics and Engineering of Photovoltaic Conversion Technologies and Systems*. UIT Cambridge Ltd, Cambridge, England.

Sobri, S., Koohi-Kamali, S., and Rahim, N. A. (2018). Solar photovoltaic generation forecasting methods: A review. *Energy Conversion and Management*, 156:459–497.

Solar Magazine (2019). Solar monkey nominated for the deloitte technology fast 50. https://solarmagazine.nl/nieuws-zonne-energie/i19378/solar-monkey-genomineerd-voor-de-deloitte-technology-fast-50, accessed: 18.04.2020.

Solar Monkey (2019a). Improvement of yield forecasting algorithm. [Internship report] Retrieved from personal communication with Alba Alcaniz Moya.

Solar Monkey (2019b). Improvement of yield forecasting algorithm. [Internship report] Retrieved from personal communication with Annanta Kaul.

Solar Monkey (2020). Automated tagging of corrupt data of daily yield from residential pv systems. [Internship report] Retrieved from basecamp personal communication with Yves van Montfort.

Taghezouit, B., Harrou, F., Sun, Y., Arab, A. H., and Larbes, C. (2020). Multivariate statistical monitoring of photovoltaic plant operation. *Energy Conversion and Management*, 205.

Tai, D. V. (2019). Solar photovoltaic power output forecasting using machine learning technique. *Renewable Energy*, 8:793–800.

The Economist (2014). Humans and mass extinction - killing machines. https://www.economist.com/books-and-arts/2014/02/22/killing-machines, accessed: 18.04.2020.

Theocharides, S., Makrides, G., and Georghiou, G. E. (2018). Machine learning algorithms for photovoltaic system power output prediction. In *Proceedings of IEEE International Energy Conference (ENERGYCON)*, Limassol, Cyprus, 3-7 June 2018. New Jersey: Piscataway.

Tóth, B. (2018). How do forward and backward propagation work? https://tech.trustpilot.com/forward-and-backward-propagation-5dc3c49c9a05, accessed: 4.06.2020.

Visser, E. (2018). Nowcasting the photovoltaic output of small-scale solar systems in the netherlands. Master's thesis, Delft University of Technology.

Yang, D. (2019). A guideline to solar forecasting research practice: Reproducible, operational, probabilistic or physically-based, ensemble, and skill (ropes). *Journal of Renewable and Sustainable Energy*, 11.

Zhang, J., Verschae, R., and Shohei Nobuhara, J.-F. L. (2018). Deep photovoltaic nowcasting. *Solar Energy*, 176:267–276.

A LITERATURE COMPARISON

Not Enough Information (NEI)

No.	Authors	Year	Title	Analyzed Systems	Country	Input Data Timespan
1	Nailya Maitanova, Jan-Simon Telle, Benedikt Hanke, Thomas Schmidt, Matthias Grottke, Karsten von Maydell, Carsten Agert	2019	Machine Learning Approach to a Low-Cost Day-Ahead Photovoltaic Power Prediction Based on Publicly Available Weather Reports For Automated Energy Management Systems	2	Germany	113 days
2	Mohamed Massaoudi, Ines Chihi, Lilia Sidhom, Mohamed Trabelsi, Shady S. Refaat and Fakhreddine S. Oueslati	2019	PV Power Forecasting Using Weighted Features for Enhanced Ensemble Method	NEI	Australia	4 years
3	Naji Al-Messabi, Yun Li Ibrahim El-Amin and Cindy Goh	2012	Forecasting of Photovoltaic Power Yield Using Dynamic Neural Networks	2	United Arab Emirates	6 days
4	D. Anagnostos, T. Schmidt, S. Cavadias, D. Soudris, J. Poortmans, F. Catthoor	2019	A Method for Detailed, Short-Term Energy Yield Forecasting of Photovoltaic Installations	1	Germany	15 minutes
5	Andre Gensler, Janosch Henze Bernhard Sick and Nils Raabe	2018	Deep Learning for Solar Power Forecasting - An Approach Using Autoencoder and LSTM Neural Networks	21	Germany	990 days
6	Kyairul Azmi Baharin, Hasimah Abdul Rahman, Mohammad Yusri Hassan and Chin Kim Gan	2016	Short-term Forecasting of Solar Photovoltaic Output Power for Tropical Climate Using Ground-based Measurement Data	1	Malaysia	334 days
7	Emanuele Ogliari, Alessandro Niccolai Sonia Leva and Riccardo E. Zich	2018	Computational Intelligence Techniques Applied to the Day Ahead PV Output Power Forecast: PHANN, SNO and Mixed	1	Italy	580 days
8	Yuchi Sun, Vignesh Venugopal and Adam R. Brandt	2019	Short-term solar power forecast with deep learning: Exploring optimal input and output configuration	1	USA	1 year
9	Spyros Theocharides, George Makrides and George E. Georghiou	2018	Machine Learning Algorithms for Photovoltaic System Power Output Prediction	1	Cyprus	1 year
10	Dinh Van Tai	2019	Solar Photovoltaic Power Output Forecasting Using Machine Learning Technique	1	Russia	NEI

No.	Features	Data Resolution	Used Algorithms	Used Metrics	Unique Conclusions
1	air temperature, humidity, cloudiness, precipitation type, max PV power in the last 5 days	5 min	LSTM	MAE, RMSE, sMAPE	It is possible to forecast PV yield without any system parameters or irradiance. Separate models for warm and cold periods. sMAPE metric allowed cross-system comparison.
2	temperature, wind speed and direction, relative humidity, horizontal and vertical radiation, PV power generated at the same instant from the previous year and the hourly time indicator	5 min	ElasticNet, XGBoost, Random Forrest, LIME, Gaussian Naive Bayes, K-Nearest Neighbors, IANN - Importance Aided Neural Networks	RMSE, MAE, R^2,	ANN are good in medium to long term forecasting, ARMA and NARX for short term. Feature importance using model reliance class (MCR). Domain knowledge contributes to the prediction accuracy.
3	temperature, wind velocity, insolation	10 minutes	MLP, FFNN - Feed Forward Neural Network, FTDNN,	RMSE, MAE	Linear time-series forecasting methods will not suit PV-systems, as the output changes rapidly in non-linear manner.
4	DHI, DNI, module temperature, Vmpp, Impp, zenith, R/B difference and deviation, sky images, solar zenith and azimuth,	1 second	NARX, MLP	RMSE %, FSS %	Clear sky moments are trivial to model.
5	NEI	3h	MLP, LSTMs, Deep Belief Networks, AutoEncoders	RMSE, MAE, AbsDev, BIAS, Correlation	Authors aim to utilize transfer learning and combine DNNs, CNNs and LSTMs to reduce error even further.
6	GHI, GPOA, module temperature	5 min	NAR- Nonlinear Autoregressive Network, SVR	R^2, MAE, RMSE %, Forecasting Skill	Model validation for different weather conditions. GPOA provides better results than GHI.
7	ambient temperature, GHI, normal irradiation on PV cell, wind speed, and direction, pressure, precipitation, cloud cover and cloud type	1 h	PHANN - Physical Hybrid Artificial Neural Network, SNO - Social Network Optimization and Mixed, MLP,	RMSE, NMAE %, EMAE %, nRMSE %, s%	Usage of physical models: Shockley diode equation, KCL, Ohm's law in combination with ANN.
8	sky images, neighboring PV module output	1 min	CNN, Linear Regression	RMSE [kW],	None of the described models exceeded 15.7% forecast skill
9	sky images, incident global irradiance, relative humidity, wind direction and speed, ambient temperature, system: Impp, Vmpp, Pmpp, other: sun azimuth and elevation,	NEI	ANN - Feed Forward NN (FFNN), SVR, Regression Trees (RT)	MAE, MAPE, RMSE, nRMSE, SS	Very low MAPE equal 0.6%. AAN outperformed other models. All models achieved skill score SS between 86% and 92%.
10	module surface temperature, irradiance, wind speed, power output	NEI	ANN, SVM, MLR, Adaptive Neuro-Fuzzy Inference Systems (ANFIS)	APE	The ANN model outperforms all other investigated techniques.

B FEATURE IMPORTANCE

In this appendix feature importance graphs for ElasticNet, Random Forest and XGBoost are presented. Feature selection of the latter for hourly models was described in chapter 4, hence here only the daily graph is presented. Please note that feature importance of Polynomial Regression was not plotted, as its features do not have any physical meaning. Also, after creating polynomials the model utilized more than 400 features which is difficult to present in a readable manner. Weights corresponding to each feature are visible on the right of each bar. ElasticNet and Polynomial Regression have built-in feature selection and absence of a particular feature in an importance graph means it was not selected by the models. Random Forest and XGBoost assigned weights only to the features chosen for them by the 3-fold RFE.

B.1 DAILY DATA SET

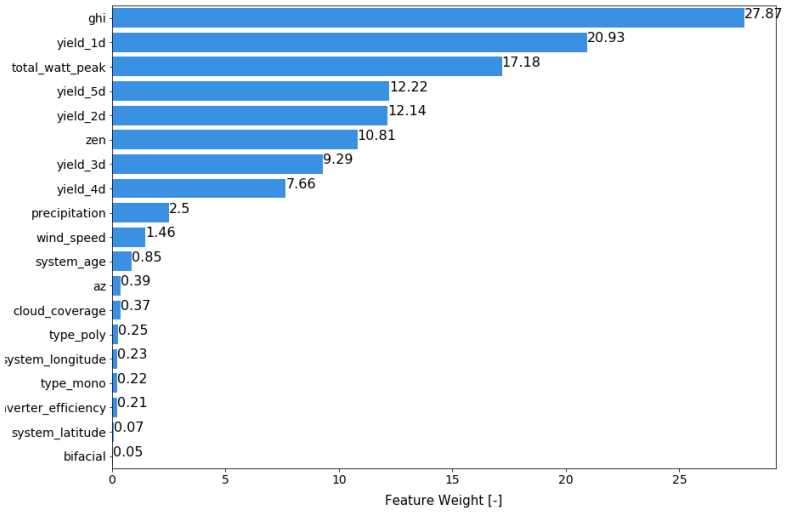


Figure B.1: ElasticNet feature importance for daily data

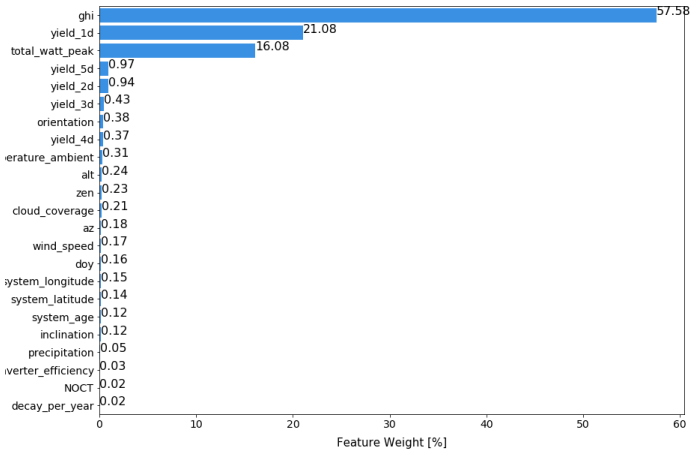


Figure B.2: Random Forest feature importance for daily data

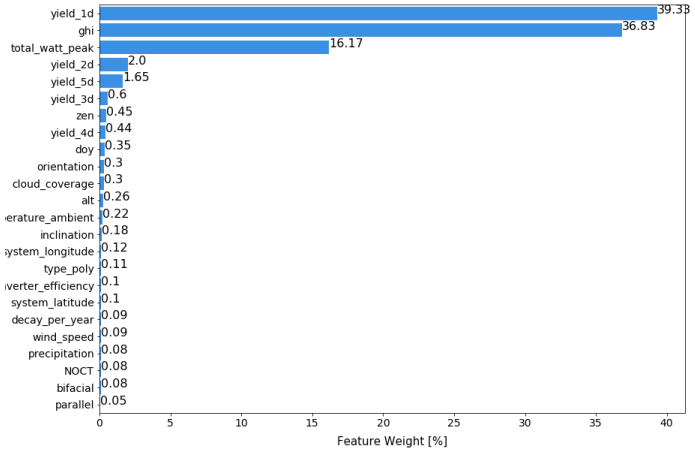


Figure B.3: XGBoost feature importance for daily data

B.2 HOURLY ROUGH DATA SET

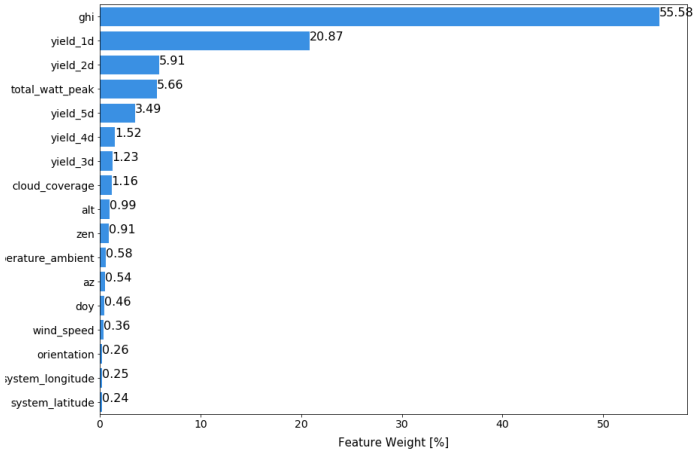


Figure B.5: Random Forest feature importance for hourly rough data set

B.3 HOURLY DETAILED DATA SET

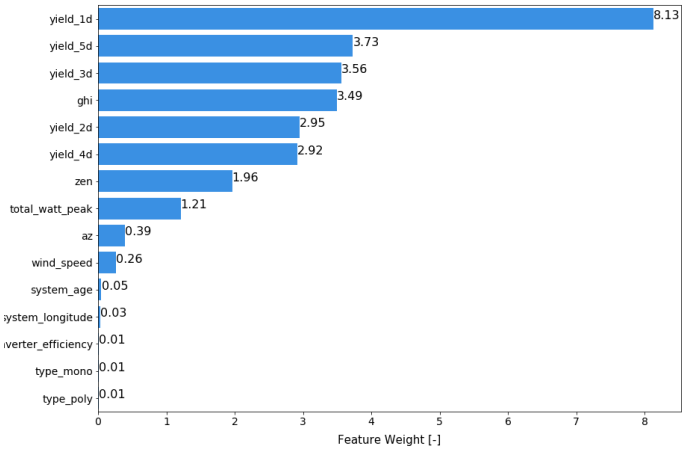


Figure B.4: ElasticNet feature importance for hourly rough data set

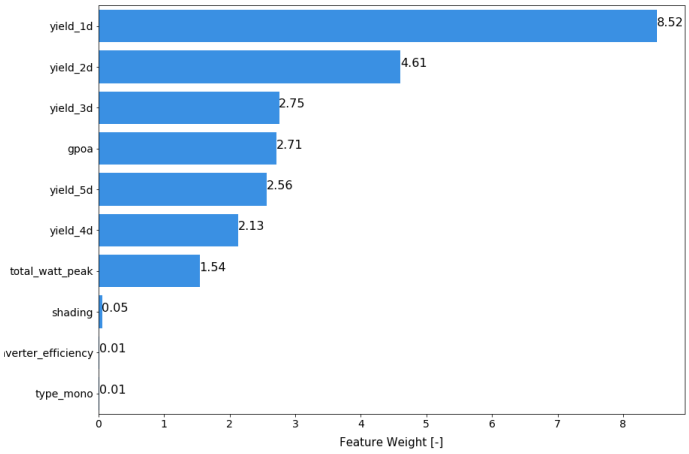


Figure B.6: ElasticNet feature importance for hourly detailed data set

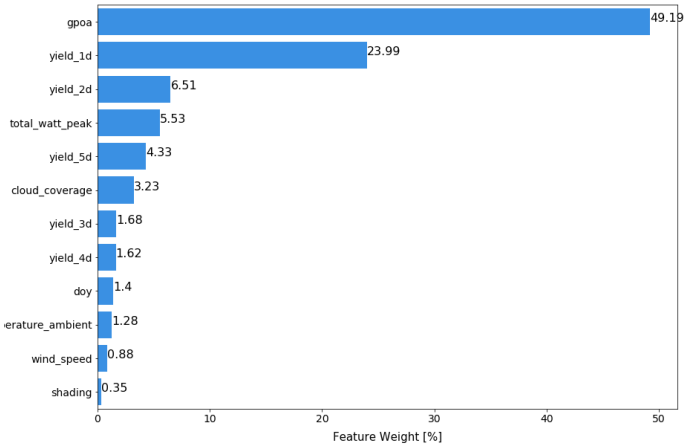


Figure B.7: Random Forest feature importance for hourly detailed data set

C OPTIMAL HYPERPARAMETER CONFIGURATIONS

In this section the results of hyperparameter tuning for ElasticNet, Random Forest and XGBoost are presented. Tuning was performed using *RandomizedSearchCV* with three fold cross-validation and 20 rounds. Better results could be obtained using manual tuning, but this path was not pursued due to large number of models and data sets. Nevertheless, the presented configurations correspond to local optima which are assumed to be close to the global optimum. It can be seen that each algorithm has different hyperparameters, but each of them corresponds either to reducing bias or to reducing variance. Their meaning is directly related to algorithms working principles described in appendix E.

C.1 DAILY DATA SET

Table C.1: Optimal configuration of ElasticNet hyperparameters - daily data

Hyperparameters	Values
l1_ratio	1.0
alpha	0.0020164908888846379
n_iter	191

Table C.2: Optimal configuration of Random Forest hyperparameters - daily data

Hyperparameters	Values
n_estimators	40
min_samples_split	3
min_samples_leaf	1
max depth	35

Table C.3: Optimal configuration of XGBoost hyperparameters - daily data

Hyperparameters	Values
objective function	reg:squarederror
colsample_bytree	1
colsample_bynode	1
colsample_bylevel	0.8
learning_rate	0.2
max_depth	40
alpha	20
lambda	10
n_estimators	50
gamma	1

C.2 HOURLY ROUGH DATA SET

Table C.4: Optimal configuration of ElasticNet hyperparameters - hourly rough data

Hyperparameters	Values
l1_ratio	1.0
alpha	0.00021539756327209055
n_iter	163

Table C.5: Optimal configuration of Random Forest hyperparameters - hourly rough data

Hyperparameters	Values
n_estimators	45
min_samples_split	5
min_samples_leaf	2
max depth	35

Table C.6: Optimal configuration of XGBoost hyperparameters - hourly rough data

Hyperparameters	Values
objective function	reg:squarederror
colsample_bytree	1
colsample_bynode	1
colsample_bylevel	0.8
learning_rate	0.2
max_depth	60
alpha	10
lambda	30
n_estimators	40
gamma	0

C.3 HOURLY DETAILED DATA SET

Table C.7: Optimal configuration of ElasticNet hyperparameters - hourly detailed data

Hyperparameters	Values
l1_ratio	1.0
alpha	0.0001977218661344080 2
n_iter	36

Table C.8: Optimal configuration of Random Forest hyperparameters - hourly detailed data

Hyperparameters	Values
n_estimators	60
min_samples_split	5
min_samples_leaf	2
max depth	40

Table C.9: Optimal configuration of XGBoost hyperparameters - hourly detailed data

Hyperparameters	Values
objective function	reg:squarederror
colsample_bytree	1
colsample_bynode	1
colsample_bylevel	0.8
learning_rate	0.2
max_depth	60
alpha	10
lambda	30
n_estimators	40
gamma	0

D LEARNING CURVES

In this section learning curves for models other than XGBoost using hourly rough features are presented. Also, the descriptions of the learning process and reasoning behind selecting algorithms are provided.

D.1 DAILY DATA

Learning curves for ElasticNet can be seen in figures D.1 and D.2. Figure D.1 depicts the initial training phase when the model has seen relatively few training examples. At first, the validation error rapidly decreases and the training error rapidly increases, as described in chapter 3. Both curves seem to stabilize around RMSE of 4 kWh. However, in figure D.2 it can be seen that further training takes RMSE down to around 3.325 for the validation set. That is aligned with RMSE for the ElasticNet in table 4.1. Based on figure D.2 it can be concluded that ElasticNet saturates after seeing around 30% of the data set and further training does not bring any substantial improvement in its predictions. This algorithm has low variance, as training and validation errors are close to each other. At the same time it suffers from high bias, as RMSE of 3.325 kWh is significantly higher than RMSE of Solar Monkey software. This was expected, as ElasticNet is a simple algorithm unable to learn non-linear patterns. Therefore, Polynomial Regression model was created next. Polynomial Regression learning curves can be seen in figure D.3 which re-

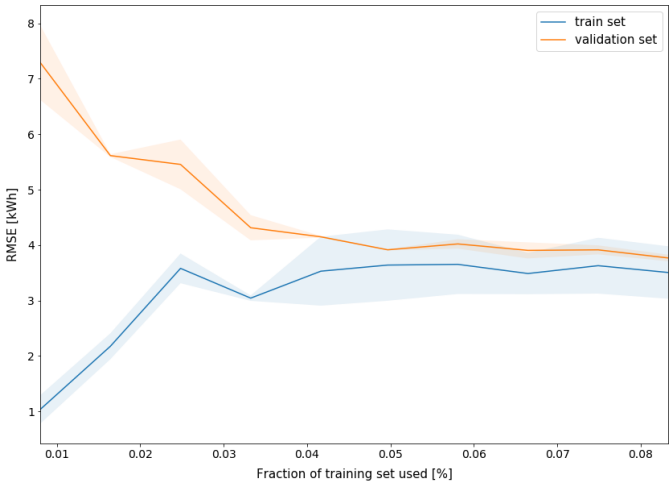


Figure D.1: ElasticNet learning curve - daily data set

veals that this model saturates after seeing around 0.5 % of the training set. When plotted for full range, the learning curves of Polynomial Regression are two constant parallel lines which imply there is no improvement with the increased training set size. The learning curves plotted for Random Forest algorithm can be seen in figures D.4 and D.5. Again, improvement in performance becomes slower with the increase in the training set size. In that sense machine learning is similar to human learning, as humans learn quickly when they know little, and when proficient in a certain domain it is difficult for us to improve further. In figure D.4 Random Forest

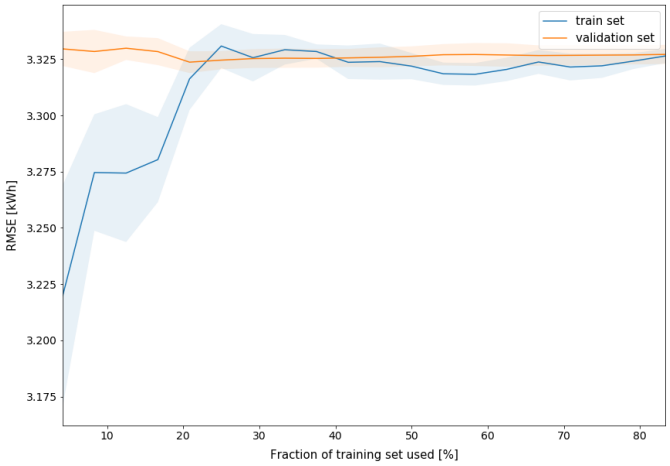


Figure D.2: ElasticNet learning curve - daily data set

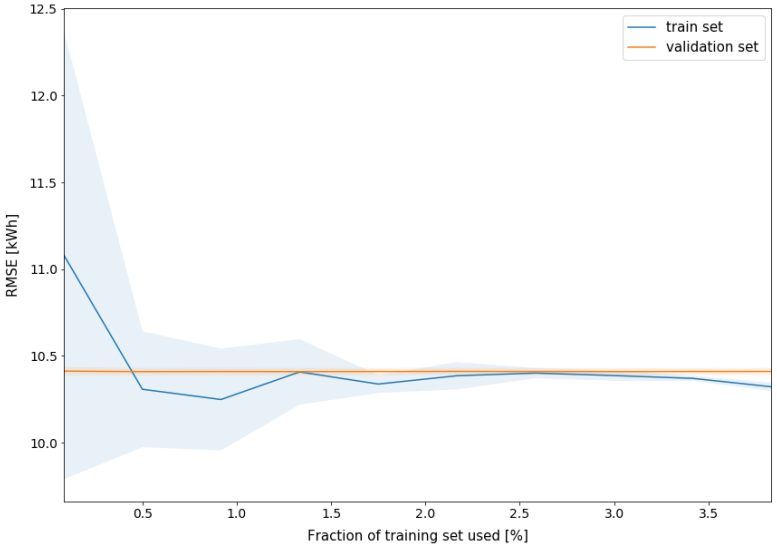


Figure D.3: Polynomial Regression learning curve - daily data set

suffers both from high bias and high variance, but it improves its performance with further training. From figure D.5 it can be concluded that its learning process saturates around 50% of the entire data set. Similarly as in chapter 4, it is merely an assumption and further training would cause decrease in error. This gain however comes at disproportionately high cost. Despite decreasing bias, at the end of the training process, the model still suffers from high variance, as validation RMSE is around 1.5 kWh and the training RMSE is around 0.6 kWh. That means Random Forest overfits to the training data and does not generalize well. Therefore, manual tuning focusing on techniques lowering variance should be performed. However, due to limited amount of time and focus on models utilizing hourly data, manual tuning was not performed. Promising results of Random Forest caused another ML ensemble model, XGBoost, to be developed.

The learning curves of XGBoost are presented in figures D.6 and D.7. Similarly to Random Forest XGBoost for daily data has relatively low bias and it can be seen that XGBoost also suffers from high variance. Several attempts of reducing it were made, but none of them lead to lower validation error. At certain point decreasing variance lead to higher bias, therefore making it impossible to improve model performance further.

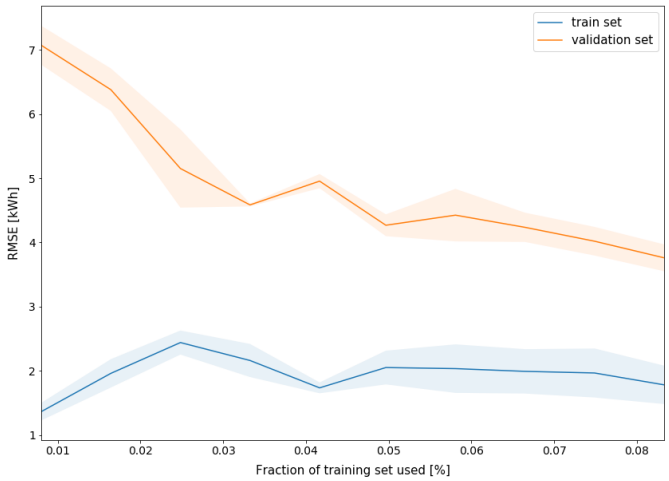


Figure D.4: Random Forest learning curve - daily data set

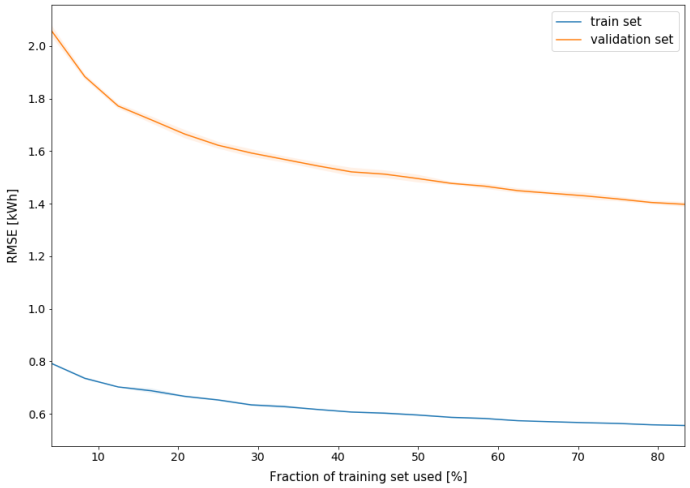


Figure D.5: Random Forest learning curve - daily data set

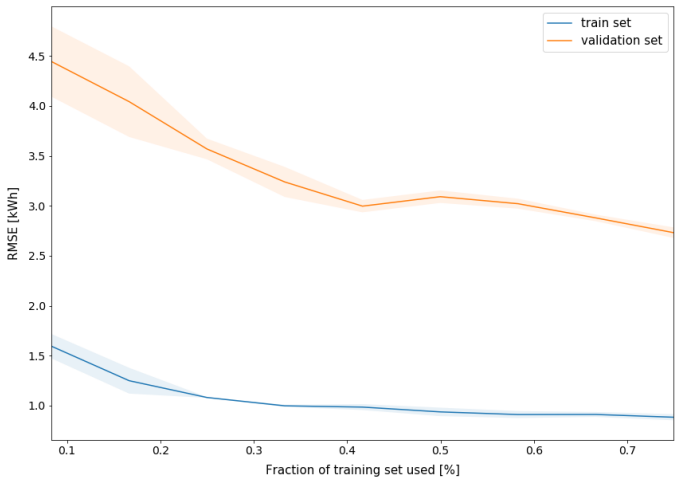


Figure D.6: XGBoost learning curve - daily data set

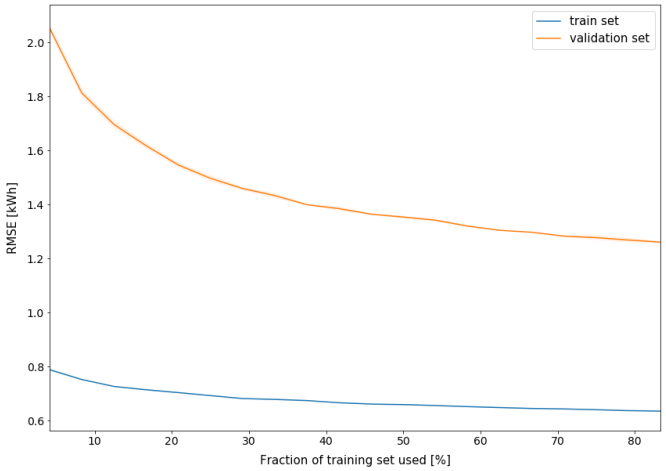


Figure D.7: XGBoost learning curve - daily data set

D.2 HOURLY DATA

Only the learning curves for the data set without GPOA pre-calculation are presented, as they are similar to those created for data including GPOA. In this case change in the learning curves is mostly caused by higher data granularity, rather than the change in feature configuration. Before the hourly graphs are analyzed it should be noticed that increased data resolution means increased data set size. Lower fraction of data set which causes learning saturation is usually caused by increased number of samples. Moreover, it should be noticed that RMSE for daily and hourly graphs should not be compared, as hourly yield values are by default lower than daily yields.

Learning curves for ElasticNet trained on hourly data can be seen in figure D.8. It can be seen that hourly ElasticNet saturates after seeing around 0.1 % of the training set. This can be explained by much larger size of the hourly data set which contains more than 4 mln samples. Therefore, seeing only around 4,000 samples is sufficient for ElasticNet to learn the pattern in the data. Training and validation error curves for full data set are two parallel lines and were not presented. Clearly this algorithm cannot benefit from larger data set. Learning curves for Polynomial

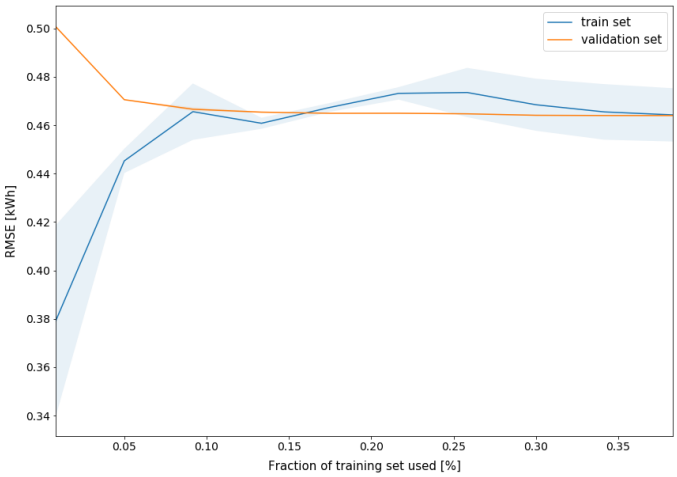


Figure D.8: ElasticNet learning curve - hourly rough data set

Regression algorithm can be seen in figure D.9. This algorithm suffers from high bias, but has low variance. Both its training and validation errors remain constant

after model being trained on 0.08 % of data. Hourly Random Forest presents similar

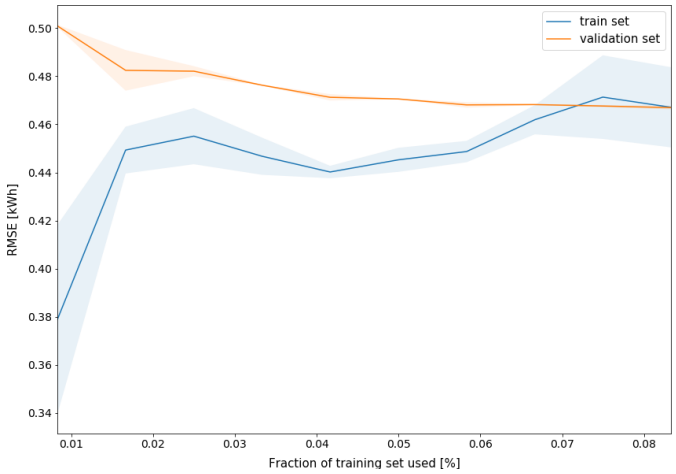


Figure D.9: Polynomial Regression learning curve - hourly rough data set

behavior to its daily counterpart. Model has low bias, but suffers from high variance. It is difficult to estimate when it saturates, as learning curve for the entire data set was not plotted due to memory issues. Nevertheless, it is expected to resemble the one plotted for the hourly XGBoost. In this section learning curves for all four de-

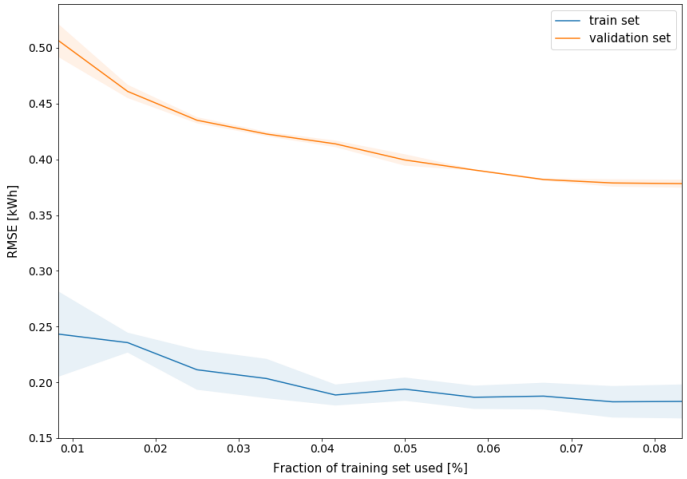


Figure D.10: Random Forest learning curve - hourly rough data set

veloped machine learning models were presented. They justify model development process, as first simple models were developed and when they did not perform well, more complex approaches were pursued. It can be also concluded that large data set available in this project was not fully utilized, as many algorithms would perform nearly as good with much smaller training set size. This has important implications for models ability to learn and was described in chapter 6.

E WORKING PRINCIPLES OF SELECTED MACHINE LEARNING MODELS

This appendix provides deeper insight into the working principles of ML algorithms utilized in this project, SVR and LSTM. It focuses on qualitative description, but provides also some information about the underlying mathematics.

E.1 PERSISTENCE METHOD

Before ML-based models are described, simple baseline *persistence model* is presented. It is not a machine learning technique, therefore it does not require any of the operations described in chapter 3. Persistence method assumes that the same conditions (yield) observed at time $t-1$ persist at time t. The model is described by equation E.1.

$$P_{p,t+\Delta t} = P_{m,t} \tag{E.1}$$

where $P_{m,t}$ is power corresponding to time sample t and $P_{p,t+\Delta t}$ is a power forecast at the moment t within time range Δt. For hourly data it is assumed that at certain time instance yield is exactly the same as it was one hour before. In case of daily data 24 hours are assumed. This model is called *simple persistence* or *naive persistence* because of its simplicity. Several other persistence models exist, some of them utilizing *clear sky index*. Their thorough description can be found in [Antonanzas et al., 2016] and [Baharin et al., 2016].

E.2 ELASTIC NET

ElasticNet is a form of linear regression, the simplest model which from the beginning was not likely to fulfill the requirements of this project. According to [Al-Messabi et al., 2012] "linear time-series forecasting methods will not suit PV systems as the output is changing rapidly in non-linear manner". However, it was built to get acquainted with *scikit learn* library, because it is easy to develop, fast in training and provides a benchmark for the more complicated algorithms. ElasticNet differs from classical Linear Regression by containing *Lasso Regression* (L1) and *Ridge Regression* (L2) regularization components. Linear regression always aims to minimize the cost function represented by formula 3.2. This equation can be modified to prevent overfitting by adding one of the two regularization terms, L1 (Lasso) or L2 (Ridge).

E.2.1 Ridge Regression

Ridge regression is represented by formula E.2 where the main difference with respect to equation 3.2 is the penalty term added at the end. Due to its presence L2 regularization influences the magnitude of coefficients, penalizing them when their magnitudes are high.

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - \sum_{j=0}^{p} w_j * x_{ij})^2 + \lambda \sum_{j=0}^{p} w_j^2 \tag{E.2}$$

E.2.2 Lasso Regression

Lasso Regression is described by equation E.3. Its only difference with respect to Ridge is that *magnitudes of coefficients* instead of their *squares* are taken into account. L1 regularization often decreases weights of some features nearly to zero.

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - \sum_{j=0}^{p} w_j * x_{ij})^2 + \lambda \sum_{j=0}^{p} |w_j| \tag{E.3}$$

E.3 POLYNOMIAL REGRESSION

Polynomial regression is not a separate algorithm itself, but a transformation of the input X matrix. *sklearn.preprocessing.PolynomialFeatures* function "generates a new matrix consisting of all polynomial combinations of the features with degree less than or equal to the specified degree" [Pedregosa et al., 2011]. For example generating polynomial features of some a and b would create features: 1, a, b, a^2, ab and b^2. In this case the specified degree was 2, as obtaining results for higher degrees was impossible due to hardware constraints. Different models can be fit to such transformed data set, but in this study ElasticNet was chosen. The main reason for that is model complexity would increase drastically if other, non-linear model was trained on polynomial features.

E.4 SUPPORT VECTOR REGRESSION

A model based on SVR was developed, but was not validated. The reason for that was its very long computational time. Training on 1,000 out of all 400,000 training examples from the daily data set was performed and algorithms convergence took around 12 minutes. This time could be significantly decreased by utilization of unsupervised learning algorithms such as PCA. However, considering the size of the entire database available in Solar Monkey, this would still be insufficient. Long computational time issue was encountered also by [Visser, 2018], therefore this model was abandoned before its validation was performed. Nevertheless, many researchers use SVR and its working principle is provided in this section.

SVR operation is based on fitting a line to a given data set in a way minimizing the distance between training points and the line. Fitting is performed based on an arbitrary tolerance called *margin* and expressed as ϵ. All points lying within ϵ from the fitted line are called *support vectors* and help in determining the closest match between the data points and the function used to represent them. However, not all data points lie within ϵ range from the fitted line. In order to account for them *slack variables* ξ and $\hat{\xi}$ are introduced. They are a part of the minimization problem formulated in equation E.4, as occurring outliers should be possibly small. Importance of outliers can be incorporated by cost parameter C (C > 0) which determines sensitivity to noise in the data. Optimization problem solved by SVR is described by equations E.4, E.5 and E.6 where ξ and $\hat{\xi}_i$ are greater or equal zero. Constraints limit the amount of SVR absolute error to an arbitrary number which distinguishes this algorithm from others. Note that in case of solar yield nowcasting for multiple systems, setting relative rather than absolute error would be more beneficial.

$$min(C\sum_{i=1}^{n}(\xi_i + \hat{\xi}_i) + \frac{1}{2}||w||^2) \tag{E.4}$$

$$\epsilon + \hat{\xi}_i >= y_i - f(x_i) \tag{E.5}$$

$$\epsilon + \xi_i >= f(x_i) - y_i \tag{E.6}$$

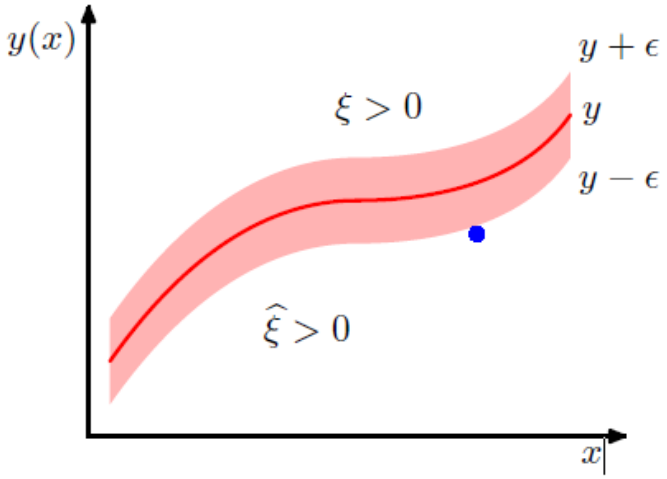


Figure E.1: SVR working principle [Bishop, 2006]

Figure E.1 represents a two-dimensional problem, so it can be visualised. However, SVR can solve also non-linear problems and its ability to do so is determined by *kernel*. Kernels are functions used to map low-dimensional data to higher dimensions and are another optimization parameter, as kernel choice is data set specific [Sharp, 2020].

E.5 RANDOM FOREST

Random Forest is built of Decision Trees which are supervised learning algorithms used both for classification and regression. They are based on splitting observations into possibly homogeneous subsets. Random Forest operation is well described by the below quote:

> *The possible solutions to a given problem emerge as the leaves of a tree, each node representing a point of deliberation and decision* - Niklaus Wirth [Esposito and Esposito, 2020]

Random Forest is an *ensemble* method which means it uses multiple models with high bias or high variance, so called *week learners* to obtain the final model with low bias and low variance, a *strong learner*. The results from weak learners are averaged to create a single, final prediction. In order to do that fitted models should be independent which is not possible when using just one data set. Therefore, n samples are randomly drawn with replacement. Underlying assumption is the samples are representative, as the original data set is large enough to capture pattern and to ensure that samples are not too correlated. **Please note that samples independence is an approximation and the drawn examples are "almost-independent" and "almost-representative"**. In other words, multiple small, artificial data sets are prepared based on the original data set. Then multiple decision trees are fit to them and their results are averaged. Usage of multiple data sets favors parallel computing making this method computationally efficient.

Weak learners used by Random Forest are *Decision Trees*. They split training data sequentially into unique regions. At every node a decision is made which splits the

training data into smaller subsets. When data at certain *node* cannot be split anymore, that point is called a *leaf*. In order to reduce overfitting *pruning* can be used. This technique reduces size of the trees by removing sections which provide little gain in the predictive power. The same effect can be obtained using *early stopping* which is a condition that interrupts model fitting. **Random forest decisions are interpretable** and help to develop understanding of the analyzed data set [Rocca, 2019], contrary e.g. to neural networks.

E.6 EXTREME GRADIENT BOOSTING

Similarly to Random Forest, Gradient Boosting is an ensemble method based on decision trees. At every iteration another decision tree is created and added to the model. However, contrary to Random Forest, in Gradient Boosting individual models do not use random subsets of data, but are built sequentially by increasing the weight of data points with wrong predictions in the previous rounds. This way algorithm gives more attention to predictions which had the largest error in the previous round. Every iteration benefits from the previous one through *boosting*, that is through changing the approach based on the results of the previous fit. New predictors learn from mistakes committed by the previous predictors which shortens the training process. Fitting is performed in a way minimizing the value of objective function and utilizing gradient descent. It can be stopped either after an arbitrary number of iterations (*boosting rounds*) or when the result of a given metric stops improving. Gradient Boosting focuses on minimizing gradient which is a first order derivative of the loss function described by equation E.7. Formula E.8 describes how predictions are updated where α is the *learning rate* and determines how big is the change in gradient at every iteration. $\sum_{i=1}(y_i - \hat{y}_i)$ depicts sum of residuals - "We are updating the predictions such that the sum of our residuals is close to zero (or minimum) and predicted values are sufficiently close to the actual values" [Grover, 2017]. Trees complexity, depth, observations per leaf and proportion of features to train on are further parameters subject to tuning.

$$Loss = \sum_{i=1}(y_i - \hat{y}_i)^2 \tag{E.7}$$

$$\hat{y}_i = \hat{y}_i - \alpha * 2 * \sum_{i=1}(y_i - \hat{y}_i) \tag{E.8}$$

Extreme Gradient Boosting is an improved version of this algorithm and calculates second order, instead of the first order, gradient and uses more advanced regularization. Other differences are mostly related to better utilization of available hardware and were described by [Morde and Setty, 2019]. XGBoost was first described by [Chen and Guestrin, 2016] who utilized their knowledge in mathematics and computer science to develop software which can fully utilize the available hardware.

E.7 GRADIENT BOOSTING WITH QUANTILE LOSS FUNCTION

Gradient Boosting with quantile loss function was developed to explore the opportunity of decreasing paid compensations to zero. Pursuing this goal, initially appealing, turned out not to match the needs of Solar Monkey, as **the monitoring software is not used for guaranteed yield predictions**. The model was created only for the daily data set without complex hyperparameter tuning and validation. Its interesting property is providing an easy tool to manipulate the peak of error distribution visible in figure E.2.

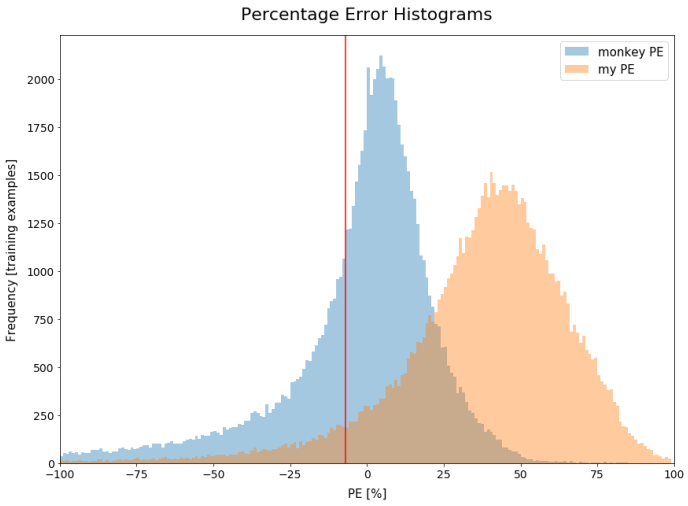


Figure E.2: Distribution of percentage error of Gradient Boosting with quantile loss function trained on data with daily resolution

An arbitrary number, called *quantile*, is an input used by Gradient Boosting with quantile loss function. The larger the quantile value, the more peak of error distribution shifts towards positive values. According to the current business model, only negative errors below -7% are compensated. The threshold is represented by the red line in figure E.2. It was assumed that having fewer negative errors would generate savings for the company. **However, that is not the case, as year-ahead predictions and monitoring are performed by two separate models**. In this particular case, predictions become more conservative, hence the model tends to under-predict. Lowering the risk of negative errors comes at a price, as the chosen metric and absolute percentage error are bound to worsen. It is important to notice that Gradient Boosting described here and XGBoost described before are two versions of the same algorithm with the latter being more computationally efficient. They utilize two different libraries, where *xgboost* library does not have built-in quantile loss function. Therefore, Gradient Boosting in *scikit.learn* was utilized.

E.8 LONG SHORT-TERM MEMORY NEURAL NETWORK

All models described in chapter 4 utilize machine learning algorithms. There is another branch of artificial intelligence called *deep learning* and focusing on applications of neural networks. These techniques significantly differ from the ones described previously, both in terms of their working principle and implementation. Most of the researchers dealing with PV yield forecasting who reported promising results utilize some kind of neural networks. These algorithms interesting property is maintaining linear learning curve even for large data sets. That is particularly relevant for this project, since it utilizes exceptionally large data set. Learning curve of the most promising machine learning technique, XGBoost, converges before reaching even 40 % of data set size (vide chapter 4) which means that Extreme Gradient Boosting would perform almost as good as reported, even on much smaller data set. Therefore, a different algorithm should be applied to fully utilize the available information and to create market advantage for the company. In this section an overview of one deep learning technique, namely *Long Short Term Memory*, including its implementation, working principle and characteristics is presented.

The mathematical theory underlying the LSTM neural networks was first described by [Hochreiter and Schmidhuber, 1997]. In order to understand LSTM working principle, a general overview of neural networks should be given. Analogies between

them and the human brain are perceived as misleading by the author of this thesis and were omitted. Neural networks consist of layers, first one containing inputs, last one containing outputs (usually one value) and an arbitrary number of hidden layers, as depicted in figure E.3. Each layer consists of nodes and both the num-

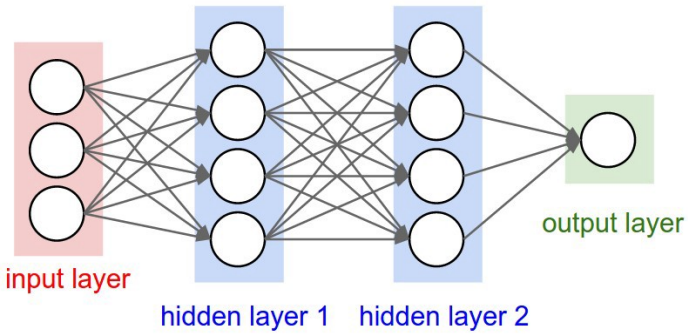


Figure E.3: Topology of a neural network

ber of layers and the number nodes are subject to optimization. Larger number of layers and nodes increases model complexity and might cause overfitting while too few of them can result in underfitting. These issues are similar to ML described in chapter 3. Firstly, neural network is initialized by an arbitrary configuration of weights and uses them to make its first prediction. Zero weights cannot be used for initialization, as this would cause gradient to remain constant and disable learning. Next, each neuron in the first hidden layer calculates the value of an activation function based on the provided features and weights. Calculated result is passed to the second layer and together with another weight is used by a node in the second hidden layer to calculate the value of an activation function. The process is repeated until the outcome reaches the output layer where it is usually combined into a single prediction. After each forward pass, the cost associated to all predictions is calculated. In the next iteration the neural network assigns weights in a way which decreases the cost. This can be performed using *backpropagation* which calculates partial derivatives of the cost function with respect to each weight, to determine which one of them increased cost the most. Such weight is changed according to the pre-determined learning rate and all process is repeated [Tóth, 2018].

How is that related to LSTM? LSTM neural networks have some additions with respect to the simple topology described in the previous paragraph. Schematic of a single node of LSTM neural network can be seen in figure E.4. Each LSTM cell has three gates: forget gate, the input gate and the output gate. Inputs come from left

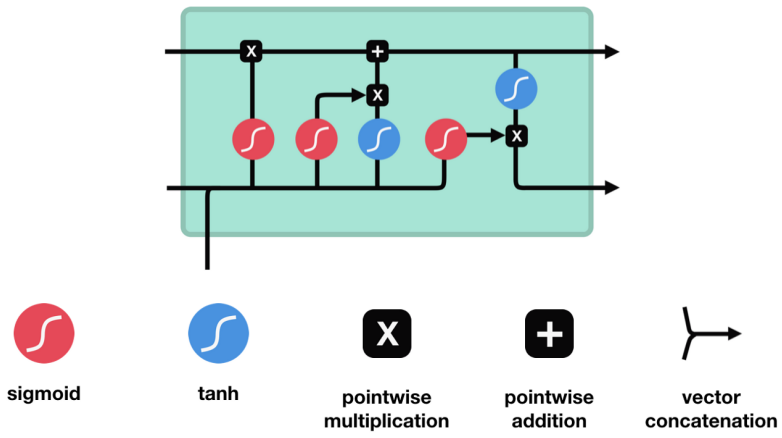


Figure E.4: LSTM cell and its operations [Phi, 2018]

bottom corner and move towards the right and upwards, going through the path indicated by the first vertical line from the left, called the *forget gate*. Inside it there is a *sigmoid function*, marked by the red circle, which can take values only between zero and one. Values close to zero correspond to "forget" and values close to one correspond to "remember". Next, the previous cell state and output of preceding LSTM cell are split and processed by two activation functions: sigmoid and *hyperbolic tangent*. Sigmoid decides which values will be updated while hyperbolic tangent helps to regulate the network. Next, the outcomes are point wise multiplied. Finally, the output gate updates cell state with the relevant values only [Phi, 2018].

Once the basic working principle of neural networks and the specific case of LSTM were described, the relevance of this algorithm for PV yield prediction can be explained. LSTMs are particularly useful for identifying patterns in PV yield data due to their memory unit. As previously described time component has paramount importance in predicting solar yield. LSTMs have the ability to remember the previous samples which can be demonstrated with a simple example of Natural Language Processing (NLP). In a sentence:

"The **cat**, which already ate, **was** full."

it is clear that "was" is dependent on whether "cat" is singular or plural. If it was plural the sentence would be:

"The **cats**, which already ate, **were** full".

LSTM can remember person of the subject used several steps (words) ago and predict the correct form of "to be" based on it [Ng, 2020]. Similarly it can memorize patterns of solar radiation and other parameters relevant for PV yield forecasting. It might be counter-intuitive that previous yields have impact on solar yield today, but feature weights assigned by all models clearly favor them. Including the time component increases predictive power of the model, but makes this approach prone to missing values.

Finally, a few practical remarks should be made, as it was attempted to build an LSTM during this project. LSTM is a time series method which makes predictions based on a window. If the window consists of 20 time steps, that means 20 measurements are required before a prediction can be made. Predictions can either refer to a single time step or multiple time steps into the future. That is a very important feature which distinguishes LSTM from other methods. It is worth noticing that monitoring is nowcasting (single step) problem, while LSTMs are suited for forecasting (multi-step) problems. However, time is relative and this constraint can be bypassed by treating present time instance as t+1 and the previous one as t. Moreover, either one (univariate) or multiple (multivariate) time series can be provided as input. Monitoring task requires single-step, multi-variate and multi-system approach which means that only one step in the future is predicted and that multiple weather and system parameters are utilized. Tensorflow LSTM model does not support a separate dimension of input data which could support plug-in of multiple systems. Therefore, time series for each feature and all systems have to be standarized and aggregated. This includes also constant system parameters such as *total watt peak* which after appending at the end of each other would create a curve similar to a step function. Aggregation can be performed using the *Flatten* layer in tensorflow. After providing these programming hints it should be mentioned that creating a LSTM neural network for 1,102 PV systems utilized in this study would require running calculations on GPUs rather than CPUs which in turn requires a different environment than Jupyter Notebooks. Google Colab which allows switching between CPU and GPU might be an interesting alternative.

To summarize, this chapter provided overview of the working principles of all ML models developed in this study extended to SVR and LSTM neural networks. While the first seven sections are significant for this project the last one provides information valuable for future researchers who would like to continue this research from the most recent point.

F EXPERIMENTAL SETUP

In order to ensure reproducibility of the results, the most often hardware configuration used in this project is presented in table F.1 together with versions of the utilized libraries in table F.2. Table F.1 refers to Google cloud's *n1-standard-4 machine* while *n1-standard-16 machine* was used as well, for hourly calculations. The main improvements in the latter are 60GB of RAM and 16 virtual CPUs compared to 15GB and 4 virtual CPUs in n1-standard-4 machine.

Table F.1: Used hardware configuration

CPU	Intel(R) Xenon(R) 2 GHz
CPU cores	2
capche size	39424 KB
RAM	15 GB

Table F.2: Versions of the utilized libraries

Library	Version
pandas	0.24.2
numpy	1.17.0
scikit-learn	0.21.3
xgboost	1.0.2
pvlib	0.6.3
matplotlib	3.1.1
seaborn	0.9.0

G | SCIENTIFIC ARTICLE

IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS

Photovoltaic Yield Nowcasting for Residential PV Systems Using Extreme Gradient Boosting (Jun. 2020)

First A. Author1, *Student Membership*, Second B. Author2, *Membership*, and Third C. Author3, *Membership*

***Abstract*—Four machine learning models for solar yield nowcasting were developed: ElasticNet, Polynomial Regression, Random Forest and Extreme Gradient Boosting (XGBoost). Utilized hourly data set consisted of data for time between July 1st, 2018 and June 30th, 2019 and corresponding to 1102 PV systems. That is five times more than the largest data set studied in the found literature. XGBoost algorithm turned out to be the most suitable for the task of PV yield nowcasting obtaining Mean Absolute Error (MAE) 0.877 kWh and Mean Absolute Percentage Error (MAPE) ... % for hourly data aggregated to daily values. Its skill score is 60.9 % with respect to simple persistence model. Metrics calculated XGBoost predictions for individual PV systems are on average two times better than for currently used commercial software. This paper exposes lack of loss function combining absolute and relative error and explains its importance for residential PV yield nowcasting and forecasting.**

***Index Terms*—forecasting, gradient boosting, loss function, machine learning, nowcasting, photovoltaics, photovoltaic monitoring, PV, solar yield prediction, quadratic loss, renewable energy, XGBoost**

I. Introduction

SINCE the industrial revolution in the XVIII-th century global population, knowledge and economy have not grown linearly anymore, but have changed exponentially. The expected human lifetime in the Netherlands has changed from 48.5 years in 1900 to 81.8 years in 2019 according to Gapminder Foundation [1]. In the same period the percentage of global population living in extreme poverty shrinked from 72% to 9% [2]. Availability of cheap and abundant energy brought unprecedented era of prosperity and welfare. Even though they were not available to all, vast majority benefited from their presence. Cheap energy largely contributed to the interconnected World we know today, with fast and affordable transportation and powerful computers that are able to both boost scientific developments and provide entertainment.

However, spectacular developments came at high cost. Emissions caused by transportation and industry owe increasing the average air temperature, leading to melting of the ice caps and rise of the sea level. Due to climate change multiple habitats were irreversibly destroyed. The amount of litter produced worldwide is so large that even if humanity disappears from Earth, signs of its presence will be still visible in soil. According to [3] another mass extinction has already started and a new geological era, the *antropocen*, has already began. Despite the efforts of European Union, most of the world is still focusing on economic growth and sustaining its basic needs, unable or unwilling to tackle the climate change on global scale. Paris agreement ratified in 2015 and signed by 176 countries aims to keep the global temperature rise below 2°C and shows that politicians have increasing awareness of the issue. However, soon after signing it one of the world's largest economies, the United States, has withdrew from the pact. It seems that media attention and political actions are disproportionate to the taken measures.

Luckily, new and promising technologies might come for rescue. Between 2010 and 2019 the market of photovoltaic modules raised by 32 % annually [4] being the largest hope to tackle climate change. With monetary impact being the crucial factor shaping human actions and policies, solar energy can have a profound impact worldwide. Solar power is abundant, affordable, easily scalable and has small CO_2 emissions, associated with manufacturing. However, mass utilization of solar modules has a major challenge of intermittency of supply, which makes it difficult to maintain power balance, to plan reserve capacity and complicates market bidding. Therefore, photovoltaic (PV) yield forecasting is an important factor facilitating energy transition and supporting investment in solar energy. Accurate forecasts decrease energy yield uncertainty, therefore reducing generation-load mismatch in the power grid and decreasing investment risk. Yield nowcasting (monitoring) ensures early anomaly detection preventing financial losses and contributing to security of PV system owners. Until recently the described tasks were difficult due to lack of suitable models. Analytical equations hold in laboratory environment, but often fail to predict yield

Manuscript received Month xx, 2xxx; revised Month xx, xxxx; accepted Month x, xxxx. This work was supported in part by the xxx Department of xxx under Grant (sponsor and financial support acknowledgment goes here).

(Authors' names and affiliation) First A. Author1 and Second B. Author2 are with the xxx Department, University of xxx, City, State C.P. Country, on leave from the National Institute for xxx, City, Country (e-mail: author@domain.com).

Third C. Author3 is with the National Institute of xxx, City, State C.P. Country (corresponding author to provide phone: xxx-xxx-xxxx; fax: xxx-xxx-xxxx; e-mail: author@ domain.gov).

in the field, with insufficient information or with large data resolution. Taking continuous measurements of all required parameters in situ is not a common practice due to high associated costs. With insufficient data it is the emergence of machine learning (ML) techniques which allowed the creation of more accurate and precise models.

Despite overwhelming abundance of literature on solar yield nowcasting and forecasting, little progress is being done. According to [5] it is difficult to compare results due to lack of standard benchmark, lack of open-source access to the utilized data, evaluation on small data sets and intentional hiding of the shortcomings. There is no consensus regarding methodology and data resolution. Available research rarely covers multiple PV systems and in the analyzed articles only [6] provided analysis for 202 rooftop PV systems while no other studies utilized data for more than 21 PV systems. Issue of small data sets was noticed also by [7]. Most of the researchers motivate their study by contributing to improved generator dispatch, power quality effects mitigation, and reducing secondary reserve capacity [7], but if their results are not validated for multiple systems, they are not reliable enough to upscale them. This study aims to fill this literature gap by providing analysis of one year data for 1102 PV systems obtained from Solar Monkey startup. Also, multiple hints for developing a standard metric are given.

II. Data

In this section, origin, main characteristics and quality of input data are explained. Inputs are divided into weather and system features and descriptive parameters. Not all features were used to make predictions, as some were rejected during feature selection. The best configuration was used for prediction of PV power output, defined as a *target value*. All features are available in the period of time from July 1st 2018 until June 30th 2019 for 1102 PV systems located in the Netherlands and Belgium. This data set is explored extensively to determine main data quality issues.

A. System Parameters

System parameters include: total system size and age, type of panels (mono, poly or thin film), latitude, longitude, panel inclination, orientation and decay per year, maximal inverter efficiency and nominal operating cell temperature. It was discovered that nearly 79 % of systems consist of mono-crystalline, 18 % of poly-crystalline and around 3 % of thin film panels. Systems of size between 2.5 kWp and 7.5 kWp dominate, but larger systems, up to 17.7 kWp occur in the data set. None of the analyzed systems was older than four years.

B. Weather Parameters

Weather parameters include: global horizontal irradiance (GHI), cloud coverage, wind speed, precipitation, ambient temperature and ground temperature. GHI is dominated by small values below 50 W/m^2 and its maximal values do not exceed 950 W/m^2. Visualization of GHI vs. day of year and hour can be seen in figure 1. According to cloud coverage data sky is almost completely overcast for vast majority of time. This is unrealistic in the Dutch climate characterized by large wind speeds. Ambient temperatures remain between 264.2 K (-9°C) and 310.2 K (37.05 °C) which is reasonable. Rainfall is present in 13.4 % of all samples and its maximal value equals 21.6 mm per hour or 37.14 mm per day.

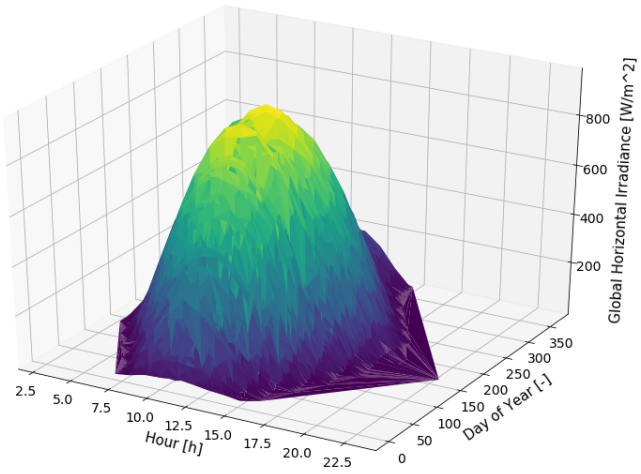


Fig. 1. Irradiance vs. day of year and hour of a day

C. Descriptive Features

Descriptive parameters consist of day of year and historical yield from 24, 48, 72, 96 and 120 hours before. Decision to use only five past days was based on [8], but a longer horizon could be utilized as well.

D. Further Data Exploration

Samples corresponding to night values (zero GHI) were removed from the data set. Missing cloud coverage values were replaced with mean cloud coverage equal to 5.83 okta. Solar Monkey conducted internal study related to the quality of yield data obtained from inverters and several issues were detected. Inverters log hourly yields via wi-fi which might be discontinuous. In case of prolonged disconnection the amount of data stored in an inverter might exceed its memory capacity and some information is lost leading to missing yield values. Other cases refer to constant or lagged yields. Constant yields might be caused by unreported disconnection of part of a PV system or prolonged partial shading. Examples of these data quality issues can be seen in figure 2. It is possible to filter out corrupt data using Hidden Markov Model (HMM), but due to time constraints existing HMM was not adjusted to hourly data resolution. Therefore, no yield data cleaning was performed.

E. Correlation Analysis

Features used as inputs to ML models should not be correlated and should have possibly large variance. These can be measured by plotting a heatmap of all input features visible

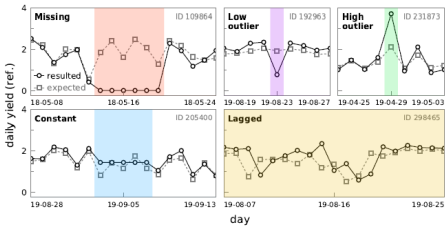


Fig. 2. Yield data quality issues CITE YVES?

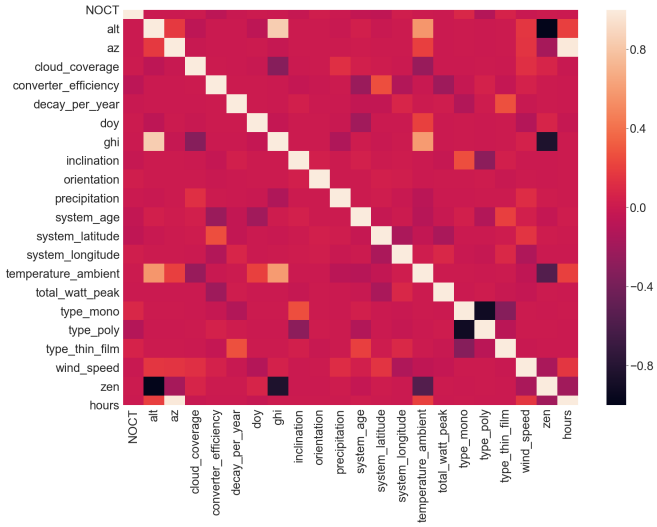


Fig. 3. Correlation heatmap

in figure 3. It can be seen that GHI is positively correlated with sun altitude and negatively correlated with cloud coverage. That is intuitive, as clouds cause shading and reduce irradiance incident on flat surface on the ground. Sun zenith has strong negative correlation with ambient temperature, as well as with GHI and sun altitude. That is understandable, as the higher the Sun zenith, the more irradiance reaches Earth surface and the hotter the air becomes. Because of these strong negative correlations solar zenith should have been removed. However, it was accidentally omitted. Ambient temperature is correlated with day of year which incorporates weather seasonality. Type poly and type mono have strong negative correlation, but it is not surprising, as particular solar panel can be of one type only. In this case however, correlation was created purposefully and does indicate these features should be removed. Also other variable not visible in figure 3, *bifacial*, was removed as it is unlikely that one out of four systems contains bifacial panels.

III. Methodology

Presented inputs had night samples removed, were processed to contain only numerical features and then standarized before being used for model training. Feature selection was performed using Recursive Feature Elimination (RFE) which determined the optimal feature configuration for the XGBoost algorithm. Next, the model was tuned for 20 rounds using *RandomizedSearchCV* with 3-fold cross validation available in *sklearn* Python library. The best configuration was trained on 80 % of randomly split data samples. Squared error loss function together with RMSE as evaluation metric were used for model training. Outcome predictions were filtered to make all negative values equal to zero.

From the beginning choice of squared error for loss function seemed to be sub-optimal. Contrary to utility scale PV forecasting, residential nowcasting requires high quality predictions for all individual systems. Squared error loss by default focuses on the largest PV systems which have the largest absolute errors. It should be noticed that developing individual models for all systems would not solve this issue, as the models would perform better in summer (higher yields) and worse in winter. Despite thorough search, none of the available libraries contains suitable loss function. MAE was rejected, because it suffers from the same issue and does not additionally penalize large errors. It is important to stress that XGBoost requires loss function which is twice differentiable. For this reason MAPE was also rejected. Huber loss becomes quadratic for small errors which makes it less sensitive to outliers and does not solve the problem neither. Log-cosh loss also allows presence of large outliers. Finally, quantile loss is used when either positive or negative errors are preferred which is not the case CITE LOSS. Several attempts to develop a custom loss function allowing to combine absolute and relative error were made, but none of them succeeded. Dividing hessian by constant system size to normalize predictions, provided disappointing results. Other approach, based on assigning weights inversely proportional to the system size to all samples failed as well. Similar is the case for model utilizing MAPE as training evaluation metric.

IV. Metrics

Extensive overview of available metrics was provided by [9] and [10] who pursued both statistical and economic approaches. They described interesting metrics such as skew, curtosis, Renyi entropy and Kolmogorov−Smirnov Integral among others. However, the most popular metrics for solar yield nowcasting and forecasting still are mean absolute error (MAE), root mean squared error (RMSE) and mean absolute percentage error (MAPE). MAE is the most intuitive, as it informs by how many kWh are the predictions off on average. That is similar to MAPE which informs about average relative (%) error. RMSE is used to penalize large errors more. It should be noticed that MAE and RMSE are absolute metrics which both increase with system size while MAPE is a relative metric. It might seem that MAPE should be the metric of choice, but it in this project it takes high values for small values of yield. An alternative can be normalized root mean squared error (NRMSE) represented by equation 1 and used also by [11]. $P_{nominal}$ corresponds to nominal PV system power while $\hat{y}$ and y correspond to prediction and observation vectors respectively and n corresponds to the number of samples.

$$NRMSE = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}}{P_{nominal}} \quad (1)$$

Other metrics used for this task are mean bias error (MBE), maximal error and skill score. Usage of neither of them is sufficient, as MBE allows negative and positive errors to cancel out, maximal error provides no information about error distribution while skill score is RMSE dependent and therefore influenced by the system size. Also, skill score informs about relative improvement with respect to persistence model which performance highly depends on data resolution. E.g. model developed in this study has skill score equal 60.9 % with respect to hourly simple persistence. If other researchers report different skill score for per minute data resolution it is likely their model is better, even if reported skill score is lower. That is because per minute persistence method is much better than the hourly one. Moreover, multiple versions of persistence model exist: simple (naive) persistence, smart persistence and persistence of cloudiness approach based on solar power index among others [9]. This makes the usage of skill score insufficient, even though it is promoted as a metric which allows comparison across projects [5]. This paper aims to expose all metric-related issues and present their impact on the quality of predictions, rather than to present the final solution of this problem.

An interesting approach was presented by [12] who managed to combine absolute and relative error in one *E-metric* described by equations 2 and 3.

$$f(\hat{y}_i, y_i) = \begin{cases} 1 & \text{if } \hat{y}_i - y_i >= p \\ 0 & \text{if } \hat{y}_i - y_i < p \end{cases} \tag{2}$$

$$E_p = \frac{1}{n} \sum_{i=1}^{n} f(\hat{y}_i, y_i) * 100\% \tag{3}$$

In equation 3 p stands for *power* and sets the threshold of absolute error. Metric E_{10} gives a percentage of predictions with absolute error below 10 Wh. Similarly E_{50} gives a percentage of predictions with absolute error below 50 Wh and so on. Its drawback is it requires different thresholds for different data resolutions. E.g. daily yields are naturally larger and if a threshold as little as 50 Wh is set, very few predictions will manage to have smaller residual error. That does not mean however that created model is low quality.

Another problem is that the majority of models is assessed on the entire test set without investigating results for individual systems. That is acceptable in case of forecasting performed for utility scale companies which operate the grid and do not need to know individual PV systems behavior. However, solar yield forecasting is becoming increasingly important for residential PV owners who would like to use it for optimizing their own production and consumption. E.g. they could store solar energy and sell it to the grid during peak hours to maximize their profits. Also, early anomaly detection requires precise and accurate yield nowcasting for individual PV systems. General metric calculation might hide cases for which the model performs badly and does not include the fact that each system usually belongs to a different entity. In case of Solar Monkey **measuring overall model performance is informative, but insufficient to determine whether all customers receive predictions of high quality**. Therefore, each metric was calculated per system and stored in an array. Next, minimum, maximum, mean and standard deviation of each array were calculated. This approach allows also to identify the worst performing systems and narrows the scope of error analysis.

V. Results

In this section values of calculated metrics for all predictions and for individual systems together with learning curves of the created model are presented. Learning saturation and critical mass of data are described.

A. Feature Selection

Features selected by RFE with their corresponding weights assigned by the XGBoost algorithm can be seen in table 4. Single most influential feature is GHI followed by historical yields. Among descriptive features these are total watt peak and cloud coverage which play a major role. Ambient temperature, wind speed and precipitation together with day of year and panel orientation have surprisingly small impact.

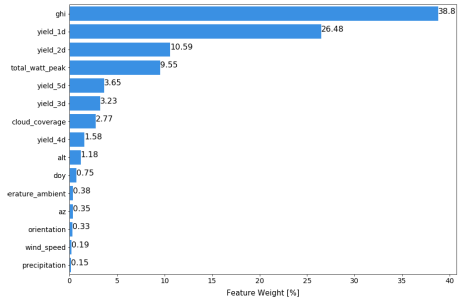


Fig. 4. Feature importance

B. Metrics Results

After performing feature selection and hyperparameter tuning for the XGBoost algorithm, its performance against the persistence method is presented in tables I and II. Next to XGBoost, performance of models based on ElasticNet, Polynomial Regression and Random Forest algorithms was presented for comparison.

TABLE I
COMPARISON OF RESULTS CALCULATED FOR ALL PREDICTIONS TOGETHER

	R^2	MAE	Max Error	RMSE	MAPE	SS
Persistence Method	0.8	0.346	8.27	0.531	70.75	N/A
ElasticNet	0.83	0.299	32.22	0.469	11176	10.88
Polynomial Regression	0.9	0.236	30.1	0.372	10504	29.33
Random Forest	0.96	0.102	28.82	0.216	417.76	59.04
XGBoost	0.97	0.102	28.9	0.206	478.01	60.9

Clearly, XGBoost provides superior results with respect to all other analyzed models. Random Forest has similar performance, but once mistaken its error is of higher magnitude, as indicated by higher RMSE and confirmed by the E-metrics in table II.

TABLE II
E-METRICS FOR HOURLY ROUGH DATA SET

	E10	E50	E100	E500
Persistence Model	10.99%	22.12%	32.85%	75.16%
ElasticNet	3.2%	15.93%	30.51%	82.1%
Polynomial Regression	4.18%	20.47%	38.18%	87.84%
Random Forest	27.43%	57.38%	72.53%	96.35%
XGBoost	23.55%	54.66%	71.35%	96.66%

Table III contains metrics calculated for individual PV systems. It can be observed that XGBoost has individual system RMSE oscillating between 0.4 kWh and 4.6 kWh. The minimal RMSE of XGBoost is similar to the one of the analytical model while the maximal RMSE for XGBoost is around two times lower. Also, mean per-system RMSE decreased by around one third. Usage of the ML model caused a drop in relative error (mean MAPE) from around 44 % to 23 % which is almost two-fold improvement. Distributions of per-system RMSE and per-system MAPE can be seen in figures 5 and 6 respectively. In both figures distributions corresponding to XGBoost model are shifted to the left with respect to the analytical model which indicates their higher quality.

TABLE III
COMPARISON OF METRICS FOR XGBOOST VS. COMMERCIAL SOFTWARE

Individual system metrics	Units	Solar Monkey	XGBoost
min RMSE	kWh	0.371	0.398
max RMSE		9.435	4.618
mean RMSE		1.824	1.355
mean MAE		1.303	0.886
mean MAPE	%	43.62	23.02
mean NRMSE	kWh / kWp	0.425	0.313

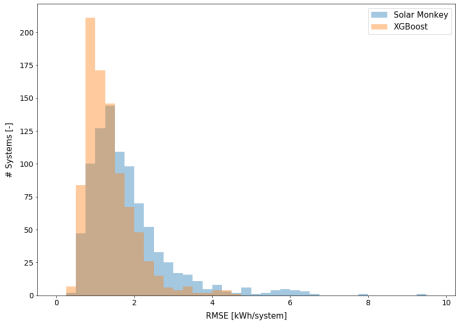


Fig. 5. Distribution of RMSE for individual systems

C. Learning curves

Next to XGBoost metrics, learning curves are presented to depict bias vs. variance trade-off and training saturation.

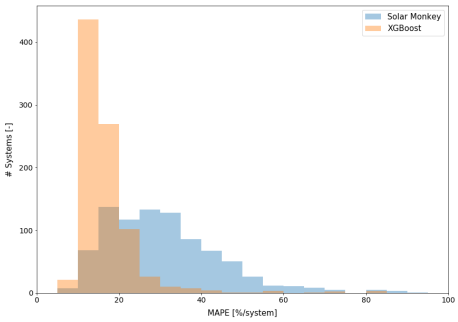


Fig. 6. Distribution of MAPE for individual systems

In this study, large data set containing more than 4,000,000 data points for 1102 PV systems was investigated. Learning curves were obtained using 3-fold cross-validation and allow to determine whether full data set potential was utilized. Semi-transparent areas around the lines correspond to standard deviations of results for all three folds. **It can be noticed that standard deviations are large for small data sets and decrease with increasing training set size**. That is particularly explicit when investigating figure 7 and is reasonable, as for large data sets it is less likely that particular shuffling would skew the result. Therefore, cross-validation can be neglected for data sets larger than 0.15 % of the training set, that is exceeding 60,000 samples, as it significantly increases computational cost and has almost no impact on the results.

Learning curve in figure 9 was plotted for custom training set sizes, which means step value was not constant, but was adjusted manually. The advantage of such plotting is both initial and final stages of model learning can be captured. Until seeing 0.15 % of the training set size XGBoost validation error decreases exponentially and it seems to stabilize later on. However, after plotting learning curve for 10 % to 80 % of the training set size it can be seen that learning does not saturate and validation error continues to decrease. However, **drop in validation RMSE is only around 0.02 kWh for training set size between 40 % and 80 %**. Therefore, it can be assumed that XGBoost learning saturates around 40 % of the training set, that is around 1.28 mln samples. In the analyzed data set each system has 4,609 samples corresponding to non-zero irradiance in a year. Dividing critical number of samples by samples per system it can be concluded that data for around 278 PV systems for an entire year is required for the XGBoost algorithm to reach sufficient quality. Further increase in training set size is likely to decrease error, but this small gain it is not justified considering surge in required computational power and the associated financial cost. Other operations, e.g. data cleaning, are likely to provide larger gain in performance at lower expense Usage of fewer number of systems with larger time horizon is likely to provide similar results, as long as the systems selected for training are representative to the test set.

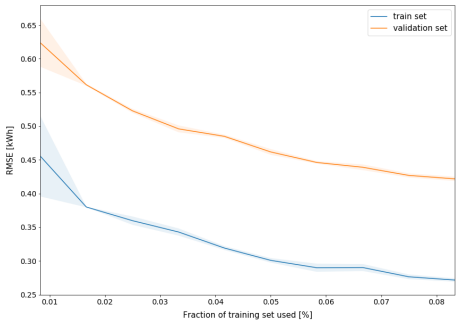


Fig. 7. XGBoost learning curve - hourly rough data set

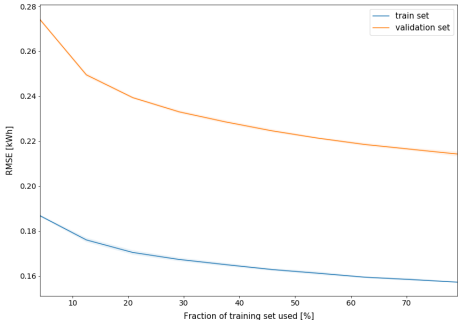


Fig. 8. XGBoost learning curve at the beginning of the learning process - hourly rough data set

VI. Error Analysis

Model performance under different weather conditions can be seen in figure 10. It can be seen it is independent of wind speed and ambient temperature, but depends heavily on GHI and to some extent on cloud coverage. Model performance under different weather conditions is presented in table IV. It can be seen that analyzed model performs around 50 % better in case of clear sky conditions which indicates that modeling clouds influence on PV behavior is challenging. It indicates also that utilized cloud coverage data is insufficient to precisely capture real life changes. Next to the weather analysis,

TABLE IV
XGBoost model performance with respect to cloud coverage

		Units	Persistence Method	XGBoost
Clear sky (<1 okta)	MAE	kWh	0.553	0.149
	RMSE	kWh	0.73	0.281
	MAPE	%	27.34	6.44
	R^2	-	0.69	0.95
Partly cloudy (1-7 okta)	MAE	kWh	0.577	0.236
	RMSE	kWh	0.768	0.377
	MAPE	%	28.64	10.73
	R^2	-	0.6	0.9
Completely overcast (>7 okta)	MAE	kWh	0.567	0.233
	RMSE	kWh	0.76	0.374
	MAPE	%	31.73	11.88
	R^2	-	0.46	0.87

error analysis was performed. It was already described that utilization of squared error loss function favours large yields, that is large PV systems and sunny hours. To further verify this

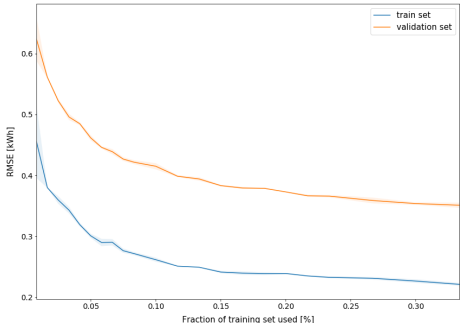


Fig. 9. XGBoost learning curve

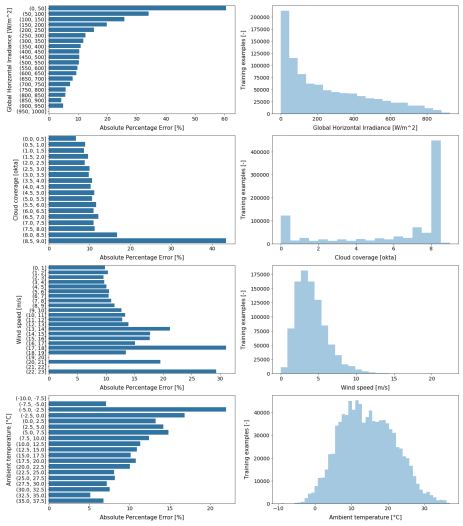


Fig. 10. Weather vs. APE analysis

hypothesis observed yields vs. their corresponding APE were plotted in figure 11. It can be seen that absolute percentage

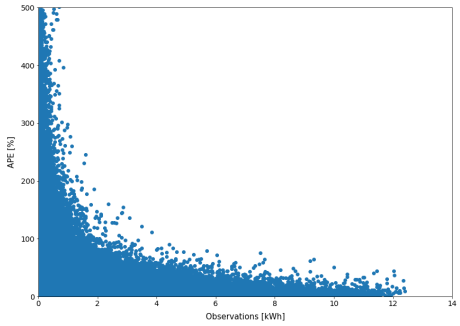


Fig. 11. Observed yield vs. absolute percentage error

error (APE) has values far exceeding 100 % for yield values below 2 kWh which confirms the initial assumption. APE with respect to hour of day and with respect to month were plotted in figures 12 and 13. It can be seen that model has the largest relative error for December followed by January. The reason for that might be that these two months have the least irradiance in the whole year and therefore the smallest yields. Analysis of APE with respect to hours has shown that

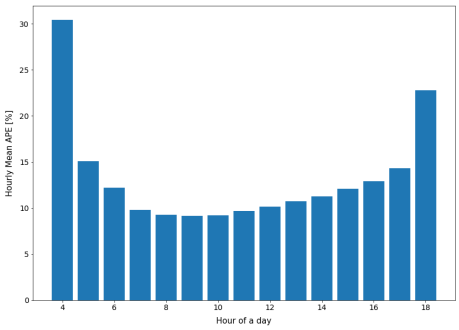


Fig. 12. Error analysis with respect to time of the day

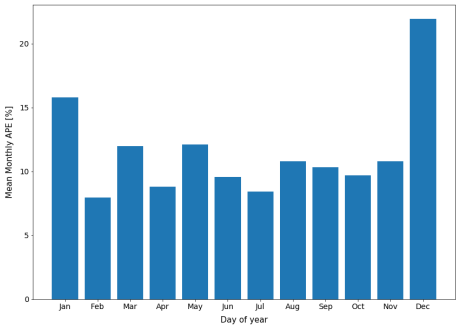


Fig. 13. Error analysis with respect to months

relative error is the largest for hours just after sunrise and just before sunset.

VII. Conclusions and Discussion

XGBoost algorithm turned out to be the method of choice for the task of solar yield nowcasting surpassing ElasticNet, Polynomial Regression, Random Forest and commercially available analytical model. This work proves that currently used metrics are insufficient for evaluation of solar nowcasting and forecasting models. Next to metrics, available loss functions fail to combine relative and absolute error which is necessary for high quality residential scale solar yield forecasting and nowcasting. Usage of squared error loss function caused the model to be drastically mistaken for small absolute values of yield occurring in winter, close to sunrise and sunset and for small PV systems. Attempt to tackle this issue through sample normalization and using MAPE as training evaluation metric have failed. Further research should focus on developing new loss function and evaluation metrics. However, despite presented issues XGBoost still provides two fold improvement with respect to commercially available analytical model.

References

[1] Gapminder Foundation, "Life expectancy in the netherlands," 2020, https://www.gapminder.org/tools/, accessed: 18.04.2020.

[2] H. Rosling, O. Rosling, and A. R. Ronnlund, *Factfullness - Ten Reasons We're Wrong About the World - and Why Things are Better Than You Think*, ser. series ?, ch. 8, pp. ?–? Leyde: Flatiron Books, Apr. 2019, an optional note.

[3] The Economist, "Humans and mass extinction - killing machines," 2014, https://www.economist.com/books-and-arts/2014/02/22/killing-machines, accessed: 18.04.2020.

[4] P. Report, "5 regression loss functions all machine learners should know," 2020, https://www.ise.fraunhofer.de/content/dam/ise/de/documents/publications/studies/Photovoltaics-Report.pdf, accessed: 12.6.2020.

[5] D. Yang, "A guideline to solar forecasting research practice: Reproducible, operational, probabilistic or physically-based, ensemble, and skill (ropes)," *Journal of Renewable and Sustainable Energy*, vol. 11, DOI XX/XXXX/ZZZ.XXXX.XXXXXXX, 2019.

[6] B. Elsinga and W. G. van Sark, "Short-term peer-to-peer solar forecasting in a network of photovoltaic systems," *Applied Energy*, vol. 206, DOI XX/XXXX/ZZZ.XXXX.XXXXXXX, pp. 1464–1483, Nov. 2017.

[7] S. Theocharides, G. Makrides, and G. E. Georghiou, "Ieee international energy conference (energycon) limassol," in *Machine Learning Algorithms for Photovoltaic System Power Output Prediction*, DOI XX/XXXX/ZZZ.XXXX.XXXXXXX, pp. 1–6, 2018.

[8] N. Maitanova, J.-S. Telle, B. Hanke, T. Schmidt, M. Grottke, K. von Maydell, and C. Agert, "Machine learning approach to a low-cost day-ahead photovoltaic power prediction based on publicly available weather reports," *Energies*, vol. 13, DOI 10.3390/en13030735, no. 3, 2020.

[9] J. Antonanzas, N. Osorio, R. Escobar, R. Urraca, F. M. de Pison, and F. Antonanzas-Torres, "Review of photovoltaic power forecasting," *Solar Energy*, vol. 136, DOI XX/XXXX/ZZZ.XXXX.XXXXXXX, pp. 78–111, 2016.

[10] S. Sobri, S. Koohi-Kamali, and N. A. Rahim, "Solar photovoltaic generation forecasting methods: A review," *Energy Conversion and Management*, vol. 156, DOI XX/XXXX/ZZZ.XXXX.XXXXXXX, pp. 459–497, 2018.

[11] J. Ascencio-Vasquez, J. Bevc, K. Reba, K. Brecl, M. Jankovec, and M. Topic, "Advanced pv performance modelling based on different levels of irradiance data accuracy," *Energies, MDPI, Open Access Journal*, vol. 13, DOI 10/3390/en.13092166, no. 9, pp. 1–12, May. 2020.

[12] E. Visser, "Nowcasting the photovoltaic output of small-scale solar systems in the netherlands," Master's thesis, Delft University of Technology, Sep. 2018.

First A. Author1 and the other authors may include biographies at the end of regular papers. The first paragraph may contain a place and/or date of birth (list place, then date). Next, the author's educational background is listed. The degrees should be listed with type of degree in what field, which institution, city, state or country, and year degree was earned. The author's major field of study should be lower-cased.

The second paragraph uses the pronoun of the person (he or she) and not the author's last name. It lists military and work experience, including summer and fellowship jobs. Job titles are capitalized. The current job must have a location; previous positions may be listed without one. Information concerning previous publications may be included.

The third paragraph begins with the author's title and last name (e.g., Dr. Smith, Prof. Jones, Mr. Kajor, Ms. Hunter). List any memberships in professional societies other than the IEEE. Finally, list any awards and work for IEEE committees and publications. If a photograph is provided, the biography will be indented around it. The photograph is placed at the top left of the biography. Personal hobbies will be deleted from the

Second B. Author2 (M'12) was born in City, Country. He received the M. degree in electrical engineering from University of City, Country in 2012.

The second paragraph uses the pronoun of the person (he or she) and not the author's last name. It lists military and work experience, including similar information to the previous author, including military, work experience, and other jobs. Job titles are capitalized. The current job must have a location; previous positions may be listed without one. Information concerning previous publications may be included.

The third paragraph begins with the author's title and last name (e.g., Dr. Smith, Prof. Jones, Mr. Kajor, Ms. Hunter), including similar information to the previous author, including the list of any awards and work for IEEE committees and publications. The photograph is placed at the top left of the biography. Personal hobbies will be deleted from the biography.

Third C. Author3 (M'99-SM'04-F'09) was born in City, Country. He received the M. and SM. and F. degrees in electrical engineering from University of City, Country in 1999, 2004 and 2009 respectively.

The second paragraph uses the pronoun of the person (he or she) and not the author's last name. It lists military and work experience, including similar information to the previous author, including military, work experience, and other jobs.

The third paragraph begins with the author's title and last name (e.g., Dr. Smith, Prof. Jones, Mr. Kajor, Ms. Hunter), including similar information to the previous author, including the list of any awards and work for IEEE committees and publications. The photograph is placed at the top left of the biography. Personal hobbies will be deleted from the biography.