Wrocław University of Science and Technology
Faculty of Information and Communication Technology

Field of study:	ITE
Speciality:	AIC

MASTER THESIS

Design and implementation of an insolation prediction system using the methods of interpolation and supervised learning

Projekt i implementacja systemu predykcji nasłonecznienia wykorzystującego metody interpolacji i uczenia nadzorowanego

Paweł Parczyk

Supervisor:

Ph.D. Paweł Zyblewski

Keywords: Supervised machine learning, Regression, Interpolation, Photovoltaic, Renewable energy sources, solar insolation forecasting

WROCŁAW 2022

Abstract

Nowadays, increasing popularity of *Renewable Energy Sources (RES)* presents us with new, previously unknown challenges. Those affect mainly in electricity power distribution infrastructure and make utility grid unstable and prone to sharp drops of current energy production. The above circumstances prompt researchers to work on systems which are able to prevent unwanted effects of energy transformation which is focused on developing more *RES*. One of considered solution for this problem could be an power production forecasting system. An approach to develop such a system is a topic of this research.

In this paper an solar insolation forecasting system is proposed. System is designed with usage of *Supervised Machine Learning* methods. Eleven different regression and interpolation models have been implemented. Performed tests allowed to determine the best regression model, which turned out to be *Mean* regression.

Streszczenie

W dzisiejszych czasach rosnąca popularność *Odnawialnych Źródeł Energii (OZE)* stawia przed nami nowe nieznane wcześniej wyzwania. Wpływają one głównie na infrastrukturę dystrybucji energii oraz czynią sieć niestabilną i podatną na gwałtowne spadki bieżącej produkcji energii. Powyższe okoliczności skłaniają badaczy do pracy nad systemami, które są w stanie zapobiec niechcianym skutkom transformacji energetycznej, która jest nastawiona na rozbudowę *OZE*. Jednym z rozważanych rozwiań tego problemu, mógłby być system prognozowania produkcji energii. Tematem niniejszych badań jest próba opracowania takiego systemu.

W tej pracy zaprezentowano system prognozowania nasłonecznienia. System został zaprojektowany z wykorzystaniem metod *Nadzorowanego Uczenia Maszynowego*. Jedenaście różnych modeli regresji i interpolacji zostało zaimplementowanych. Przeprowadzone testy pozwoliły wyznaczyć najlepszą metodę regresji, którą okazała się być regresja średnią.

Table of content

Abstract 1

Streszczenie 1

1 Introduction 5

1.1 Motivation . . . 5

1.2 Research aims and goals . . . 6

1.3 Thesis structure . . . 6

2 Selected topics of photovoltaic and machine learning 7

2.1 Photovoltaic power plant overview . . . 7

2.2 Machine learning background introduction . . . 11

3 Experimental environment and research plans 29

3.1 Programming environment . . . 29

3.2 Dataset . . . 29

3.3 Models comparison . . . 33

4 Overview of the research and tests performed 35

4.1 Various models comparison . . . 35

4.2 Decision Tree max_depth parameterization . . . 37

4.3 Random forest parameterization . . . 38

4.4 K-NN parameterization and comparison with NN interpolation . . . 38

4.5 Gravity regression parameterization . . . 41

4.6 Comparison of selected regression methods and mean regression . . 45

4.7 Normalization affect on prediction accuracy . . . 46

5 Summary, conclusions and directions for further research 49

Bibliography 51

Chapter 1

Introduction

In these days, people change their point of view on the Earth and environment. The world is beginning to see the irreversible consequences, which we are having on our surroundings. This affects many aspects of our daily lives. It has changed our habits in multiple fields. People pay attention on rubbish sorting, on what they buy and what material is it made from. It is important to buy ecological food and avoid plastic usage.

1.1 Motivation

The electricity market also cannot resists this phenomena. The industry tries decrease its affect on the environment. It changes its technology, processes, in order to be more environment friendly by decreasing pollution. Especially the pressure is laid on reducing carbon dioxide CO_2 [7] emission. This goal is being achieved by replacing fossil-fuels-based energy with more environment friendly energy sources such as wind energy, solar energy, bio-gas energy and others which are called *Renewable Energy Sources (RES)* [22].

Nowadays, according to European Union long term climate strategy European countries are going to achieve carbon neutrality until 2050 year [2]. This purpose is done in different ways in different countries. For some it is more difficult challenge and for the others is easier problem to solve. There are multiple approaches to do so, such as grants, tax facilitation and special law. In Poland this is a big challenge because there is still (data for 2021 year) more then 87% of needed energy provided from burning fossil-fuels (53,6% coal, 26,14% lignite, 7,7% natural gas) [5]. There rest of needed power is provided by different *RES* and hydroelectric power plant.

Among all types of *RES*, polish government focuses on photovoltaic energy. There are multiple reasons why this form of renewable energy is preferred. First is flexibility. Single photovoltaic power plant may have different power from a hundreds of watts to mega-watts. That feature makes it interesting for both big business and usual polish families. The second feature is an affordable price. Therefore, in 2019, the Polish government wanted to encourage almost everyone who has a house or a field to buy and install residential photovoltaic power plant, set up the first huge financial support program which was aimed to grow of the renewable energy sources share in the energy market. This program was named *Mój prąd* [8] and it is held by *National Fund Environmental Protection and Water Management* (*Narodowy Fundusz Ochrony Środowiska i Gospodarki Wodnej* [9]). In results of the program, usual polish family was able to buy and mount photovoltaic power plant, which generates enough energy to provide their annual energy consumption. Therefore in a last few years we can observe rapidly increasing amount of installed *RES* power [4] and energy produced by them [6].

1.2 Research aims and goals

In these thesis the pressure was laid on making research on prediction system of solar insolation in different places based on measurements from existing photovoltaic power plants.

Due to the possibility of pre-emptive actions tested ability would be very efficient solution for preventing problems described in *Section* 2.1.1. It allows energy provides companies to be prepared for grid instability, which probably decrease cost of maintaining system ready to be used all the time (*Section* 2.1.1).

This ability would be also beneficial to provide smarter energy management of hybrid photovoltaic solar power plants. The forecasting system may allow customer to store energy excess or export energy when needed. When system detects possible incoming over voltage, it would empty batteries which affect in avoiding export when the problem occurs and preventing turning off inverters, due to, high voltage in the grid. In the future, when energy market for residential will change into next-day market [15], the production, energy usage and price forecasting would allow customer to speculate on energy. It means to buy it when energy is cheap and sell when energy is expensive.

1.3 Thesis structure

This paper is structured as follows. In the *Chapter 1* motivation, goals and structure is described. The *Chapter 2* is a theoretical part of this paper. Residential photovoltaic system architecture and machine learning background are described there. This chapter contains information needed to understand goals of this work and methodology used in the research. *Chapter 3* is about tests. There are research environment, evaluation protocol and tests planned described. In *Chapter 4* conducted tests are described and results are analyzed. *Chapter 5* contains summary and plans for future research.

Chapter 2

Selected topics of photovoltaic and machine learning

This chapter focuses on introduce needed, in purpose of this thesis, theoretical background in photovoltaic and machine learning areas. First, photovoltaic system is described. How it works and what problems the existing architecture causes. Second, the machine learning methods and evaluation methods is described.

2.1 Photovoltaic power plant overview

Usual residential solar power plant is made of a few types of devices such as solar panels, inverters, cables and monitoring system. It contains from six to several dozens of panels with the accumulative power of 2-10 kWp. Photovoltaic panels are connected into arrays (parallel or in series). Then those arrays are connected to the inverter to maximum power point trackers (MPPTs) [1]. Inverter is also connected to the utility grid. Structure of such a system was shown in the *Figure* 2.1. The simplest system algorithm is to produce as much energy as possible when panels have been exposed to sunlight. Produced energy is consumed by the closes loads. If there is more energy than could be consumed in local system (home, office) then the energy is exported to the utility grid. (More detailed description available here [16]).

The system architecture provide many advantages for residential usage. The has is ability to store energy produced at day to the night. It is very important because consumption is rather constant over a day and night when production is only at day. An example of how energy flows in a summer day was shown in the *Figure* 2.2.

The second advantage is ability to store energy from summer to winter. It means that in summer when day is long and the weather is good installation may produce much more energy than is needed and exports it to the utility grid. In winter when day is much shorter and installation can not produce enough energy, the lack co-uld be provided from utility grid. An example of such behavior was shown in the

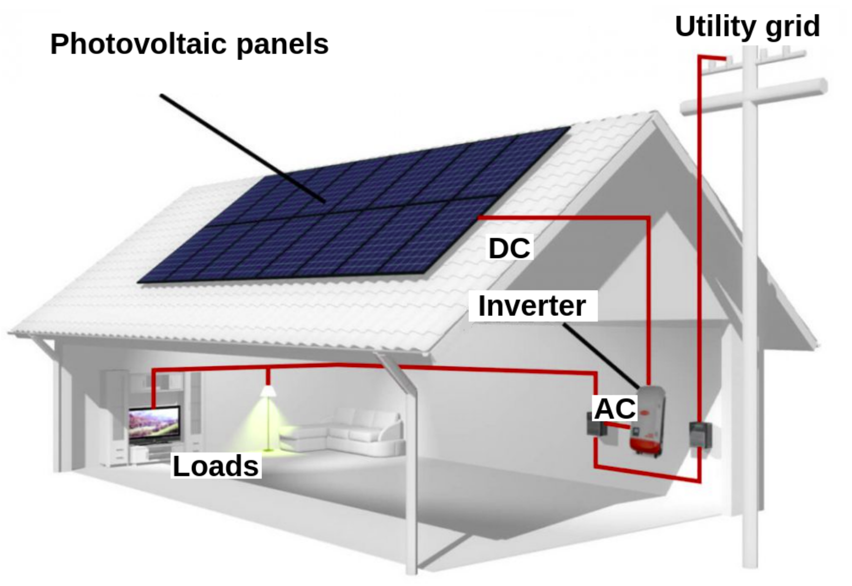

Figure 2.1 Usual residential solar power plant components [27]

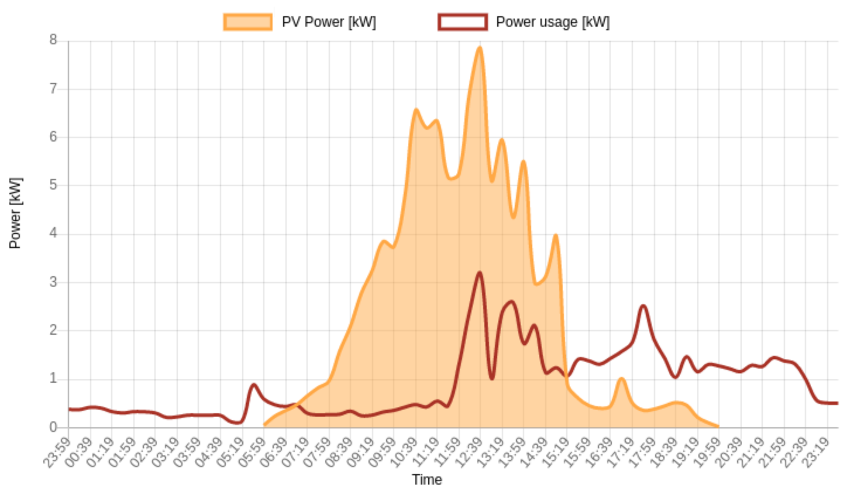

Figure 2.2 Photovoltaic power plant production and residential consumption

Figure 2.3. Energy import (red bars) is much higher than the energy export (blue bars) in winter and the opposite situation can be observed in summer.

After all, when energy provider is going to account the annual energy consumption they sum up import and export, and calculate the final price which customer has to pay. The way how to calculate it is defined in the act of renewable energy sources [18] and recently changed by [19].

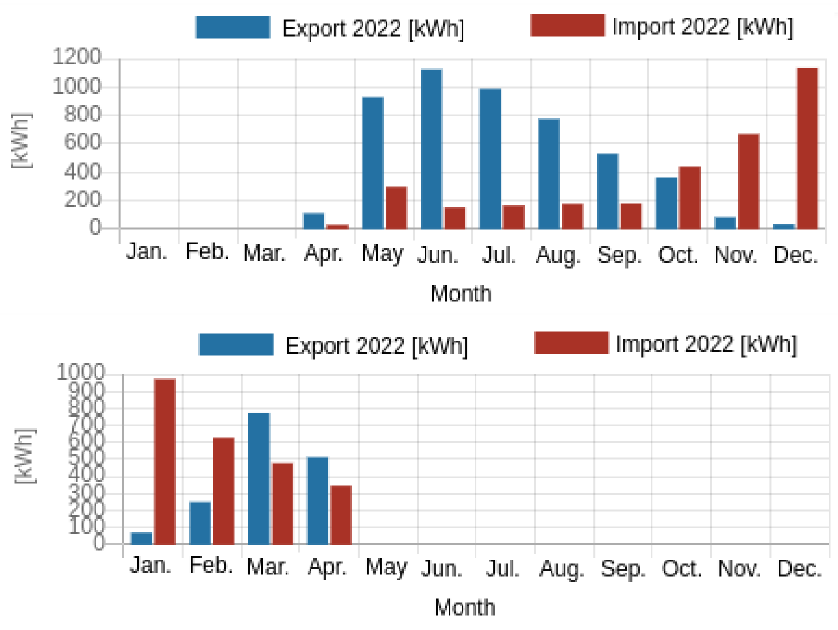

Figure 2.3 Energy storage from summer to winter

2.1.1 Problems with system architecture

The system architecture described in *Section* 2.1 was perfect for customers but when more and more installation were installed, the problems which this architecture introduced became more significant.

Those have to be solved in order to provide reliable power system.

Sometimes there is too much energy from *RES* and at the other times there is too less energy from *RES*. Both situations are undesirable.

Instability of a power generation

The first problem is instability and control impossibility of power generation. This problem occurs especially in a sunny day with some clouds on the sky (partially cloudy weather). Photovoltaic generates huge amount of energy when direct sunshine lays on the photovoltaic panels (sometimes it is a significant part of market usage), but when Sun hides behind the clouds the power generation drops drastically (*Figure* 2.5). If it drops then conventional power plants coal-based or gas-based has to fill the power supply shortage. Because of that they have to be ready, to start producing all the time which generate additional cost.

Over voltage in utility grid

The second problem is exceeding voltage limits in utility grid which prevent installation from power generation (Figure 2.4*). This problem occurs in a sunny day when irradiation is high and there is a quite low power consumption (load)

*Own study on original inverter's manufacturer monitoring hyponportal.com in a neighborhood, then due to revers power flow (energy flows from residential to utility grid) and multiple power sources (each residential photovoltaic power plan) the grid voltage increases. When voltage exceeds limit 230/400V ± 10% [17] inverters stop. This problem has been known for a long time. In 2007, in an article [29] solution was proposed. Authors conclude that a little storage system is enough to reduce voltage to prevent the inverter from turning off. Nowadays, energy storage technology was changed. Proposed acid-lead batteries can be replaced with much more efficient and safer lithium ion batteries. Systems where both photovoltaic arrays and batteries are installed are called hybrid photovoltaic systems. We can also observe changes in polish law, which makes battery systems more economically viable.

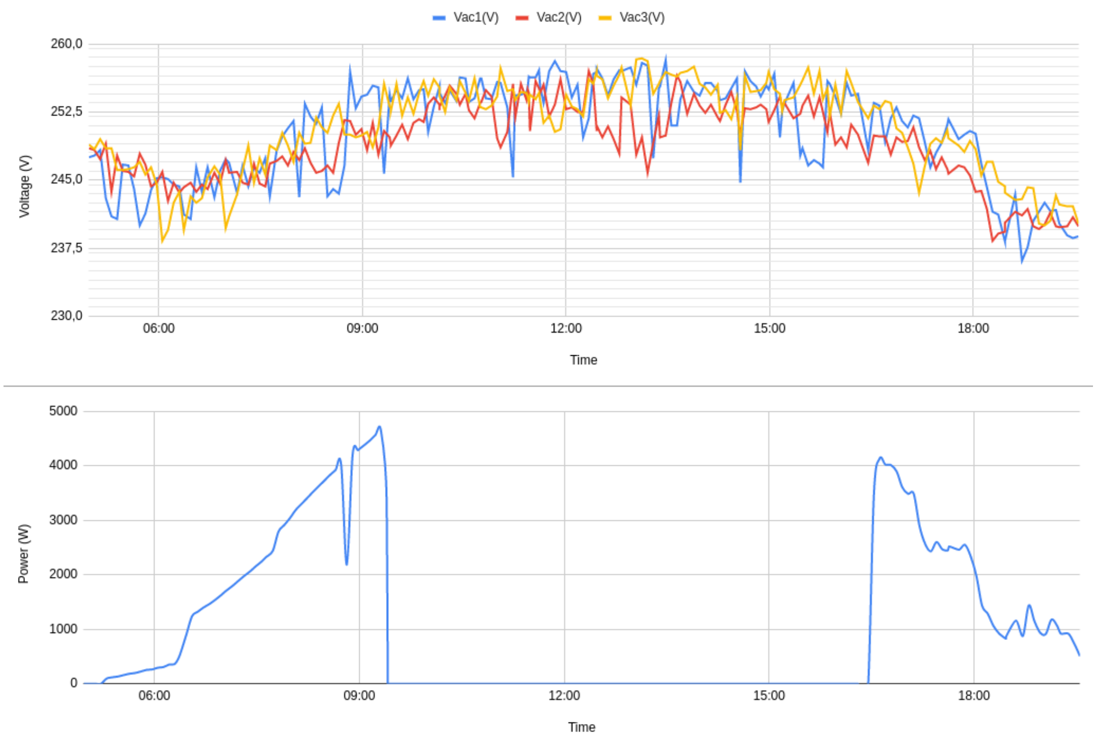

Figure 2.4 Power production interrupted due to over voltage in photovoltaic power plant located in Szewce near Wrocław at date 10-05-2022.

2.1.2 Solutions

Since many years scientists and business try to solve problems described in *Section* 2.1.1. There were many different solution proposed.

Solving the problem of instability 2.1.1 is not an easy task. The first idea is fill the lack of energy with the fossil-fuel based energy. Coal is not a good solution because coal turbines have no ability to be stopped or started in a short time. The

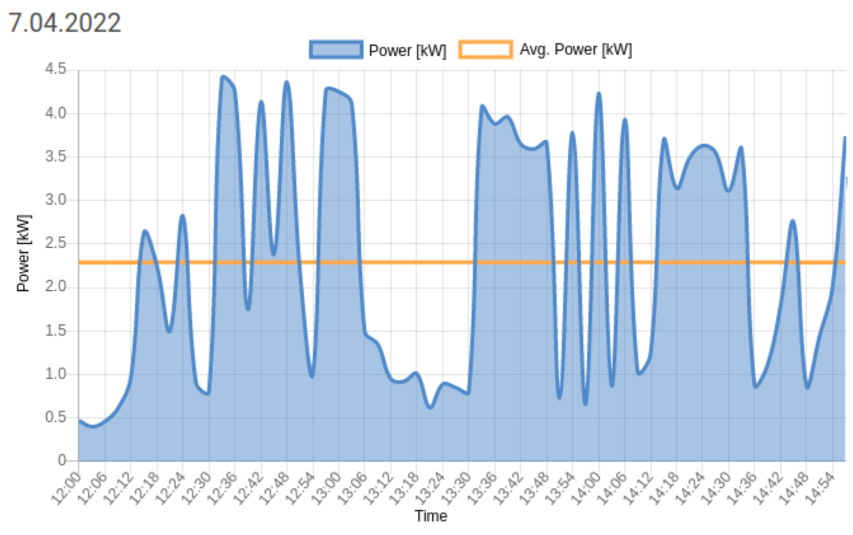

Figure 2.5 Power generation in a partly cloudy day by photovoltaic power plant (6.12 kWp)

better solution is to use a gas turbines. There is also a possibility to store energy in a big battery stations, charge them when there is more energy than needed and discharge them when there is a lack of energy.

To solve over voltage problem (*Section* 2.1.1) we have to rebuild infrastructure. In many cases problem is caused by lines, which cannot transfer more energy (i.e. the wires are too narrow and their resistance is to high for such a current). We can also force customers to maximize auto consumption and lower amount of exported energy.

2.2 Machine learning background introduction

Nowadays, computer is a powerful tool used in all fields of human life. But in order to solve a problem, apart from the computer, there is also an algorithm needed which defines what steps computer has to perform to solve such a problem. There is many approaches in algorithms development. In this work machine learning has been used.

Machine learning is an algorithmic technique where programmer do not program algorithm explicitly. The solution starts with data. Each individual in dataset is described by a *d-dimensional* feature vector x, where $x \in X$, where X is a set of all possible feature values.

$$x = \begin{bmatrix} x^{(1)} \\ x^{(2)} \\ \dots \\ x^{(d)} \end{bmatrix}, x \in X = X^{(1)} \times X^{(2)} \times \dots X^{(d)} \tag{2.1}$$

where: $x^{(l)} \in X^{(l)}$

Each feature combination belongs to the predefined category *label* $m \in M$ [34]. The data defines also a type of problem which we are going to work on. There are two types of problems.

1. **Classification** - It is a type of machine learning problem where labels take a finite *n-number* of classes $M = \{1, 2, 3, ..., n\}$.

2. **Regression** - It is an opposite problem to classification. In regression labels can be a real numbers and there is an infinite count of them $M = \mathbb{R}$.

A general use algorithm is used to build a *Model*, which is an function, which assigns features set into labels, according the Equation 2.2.

$$y = X \rightarrow M \tag{2.2}$$

Often use algorithms are *Polynomial regression*, *Support Vector Machines* or *Artificial Neural Networks*. Model is not perfect so it may make a mistake during assignment. *Metric* is a function which gives us information about this mistake. Let y be predictions made by a model on validation set and *VS* be validation set true labels, then sample metrics *Accuracy* (for binary classification) 2.3 and *Coefficient of determination* R^2 (for regression) are defined respectively by Equations 2.3, 2.4.

$$\text{accuracy}(y, VS) = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.3}$$

$$R^2(y, VS) = 1 - \frac{\sum_i (y_i - VS_i)^2}{\sum_i (y_i - \bar{y})^2} \tag{2.4}$$

The selected algorithm is learnt with learning data. The learning process is a procedure when model is adapted to the learning data. The model should find relationships between *features* and *labels*. In result, it affect in bettering obtained metric value. We distinguish a few learning methods.

1. **Supervised learning** [21] - It is a machine learning technique where programmer has data set which is consists of features and corresponding to them labels, i.e., an image classifier model is learned with an image with a different cars types with textual information what type of a car that image contains.

2. **Unsupervised learning** - In unsupervised learning model is fitted only with features, there is no labels given. The algorithm searches for distinctive features in learning data, then define groups and at the end assigns instances to the groups. This technique can be used, i.e., in clustering problem. One of the most influencing problem is to answer the question how many groups of data instances are in dataset [20].

3. **Semi-supervised learning** [33] - It is a combinations of both previous methods. In *semi-supervised learning* data set is consists small amount of labeled and big amount of unlabeled data. Labeled data are expensive, time consuming or hard to collect and it often require human annotator to label it, on the other hand unlabeled data are easy to collect. Using semi-supervised learning allow to obtain better generalization thanks to bigger and cheaper data set.

As soon as the model is fitted it can be used to perform prediction. Predictions are made on the validation data VS. It is a process when model is used to assign a new data instance into a label. Prediction may be satisfactory or not satisfactory. It depends on the problem type and expectations. In order to be able to evaluate model predictions (answering the question if model is good or bad) a special functions *metrics* are used. Metric is a function which takes predictions and correct labels as input and returns a *Metric Score*.

After the model is checked with metric we obtain a *Metric Score*, but still it is not enough to answer the posed question. The selection of learning and validation data set could have a huge influence on final result. In this moment another question could be pose. How to divide data set into learning and validation sets? Answer for that question could be find in the Article "How to design the fair experimental classifier evaluation [28]". Researchers propose *Repeated K-fold Cross-Validation* as a good solution for dataset division. Of course random division is still prone on other errors, i.e., randomly choose subsets may not be able to preserve the features of the original set.

K-fold cross-validation divides dataset into K equal subsets and uses $K - 1$ subsets for training and remaining subset for model evaluation. Repeated states for repeating K-fold cross-validation m-times. For each time division is made in a different way. It leads us to train many different models on different training sets which results in good presenting generalisation.

The result of cross-validation is a vector of metric scores. Calculating mean value and standard deviation can give the basic comparison of the models. It allow us to tell which model is better for dataset we have. Unfortunately it can not tell us if the model will be better for a whole population. In that moment statistical analysis would be helpful and enable us to answer the question "does proposed model outperform the other models?".

First, let assume that y_1 is an prediction of the first model and y_2 is an prediction of the second model then, *hypothesis* H_0 means that none of these prediction is better than the other, while *hypothesis* H_1 means that, models predict in different ways. Now, the statistical test *t-test* can be used to reject or approve H_0 or H_1.

Second stage is to calculate *probability value (p-value)*. It gives us information, what is probability that *hypothesis* H_0 is true (differences in predictions could be result of a random factor). If p-value is high, than differences between obtained results in models prediction are random. P-value goes with α factor which is a threshold level of *p-value*. If *p-value* is lower than arbitrary defined α we assume that differences in models prediction are not random and *hypothesis* H_0 is false. If we are going to compare k-models (where k is a number of models under consideration) on the same dataset we have to process $\frac{k \cdot (k-1)}{2}$ pairwise comparisons.

2.2.1 Evaluation protocol

The goal of this work is to make a comparison between different models prepared to solve the problem. To be able to do so, it is necessary, to provide functions and methods that will be able to evaluate results. The set of functions and methods is called *Evaluation protocol*and it is consists of a few steps.

1. **Data filtering** - The first step of evaluation protocol is data filtering. This step was proposed to remove a part of dataset from learning, to limit installation area.
2. **Cross-validation**.
   (a) **Data normalization** - Preprocessing step when some features are changed, i.e., latitude and longitude are scaled into $< 0, 1 >$ range. The main goal of this step is to make data better for the algorithm.
   (b) **Learning model** - A basic machine learning step where model is adopted to learning data.
   (c) **Making predictions** - after the model has been learnt, it is used to make a predictions on a validation dataset.
   (d) **Evaluation** - comparison obtained in previous step predictions with true values. Comparison is made with selected metrics.
3. **Model comparison - Statistical analysis** - Comparing different models in order to define which is better than the others.

2.2.2 Algorithms for regression and interpolation

In this section, there are presented regression and interpolation models which have been used during research. There are two groups of them. The first group is a reference group. This group is consists of Random regression 2.2.3 Mean regression 2.2.4, and Ones regression 2.2.5. They are used to determine error level for rest models, i.e., if a model is worse than random then it is definitely a bad model. Each model is an object which has to be fitted with a learning set and then it has ability to predict values regarding *Equation* 2.5. A regression model is fitted with power on a particular time of all installations. In the simplest case features are location (latitude and longitude) and labels are normalized power. History data is not used in static approach model.

$$y = predict(X\_train, Y\_train, X\_test) \tag{2.5}$$

where:

y	predicted values,
predict	model's prediction function. It is different for each model,
X_train	training dataset appointed during cross-validation.
Y_train	training true values appointed during cross-validation.
X_test	test dataset appointed during cross-validation.

Note: For purposes of this section, the data set has been limited only to the Lower Silesian Voivodeship installations according to *Figure* 2.6. Red samples were discarded from further processing. Dataset is described in *Chapter* 3.

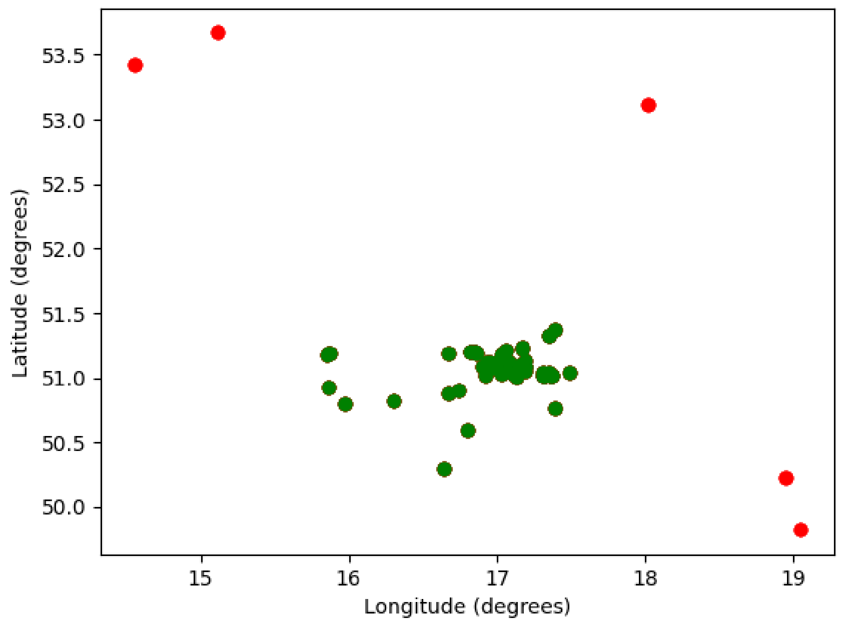

Figure 2.6 Selection of installation in order to decrease effect of uneven distribution (green - taken, red - rejected)

Note: Most of charts in the following chapter was prepared in the same way. There is mesh grid in the background and scatter plot in the foreground. The plot represents function $f(x, y) = z$, where $z \in < 0, 1 >$ The z value is expressed by *inferno* color map, but the background has *alpha* color factor set to 0,7. Scatters are the same for all charts. In the top right corner there are two numbers. The highest one is a maximum value of mesh grid and the second one is the minimum.

2.2.3 Random

Random regression draws a number from a given range for every prediction. The model works in accordance with *Equation* 2.6.

$$y = \mathrm{rand}(0, 1) \tag{2.6}$$

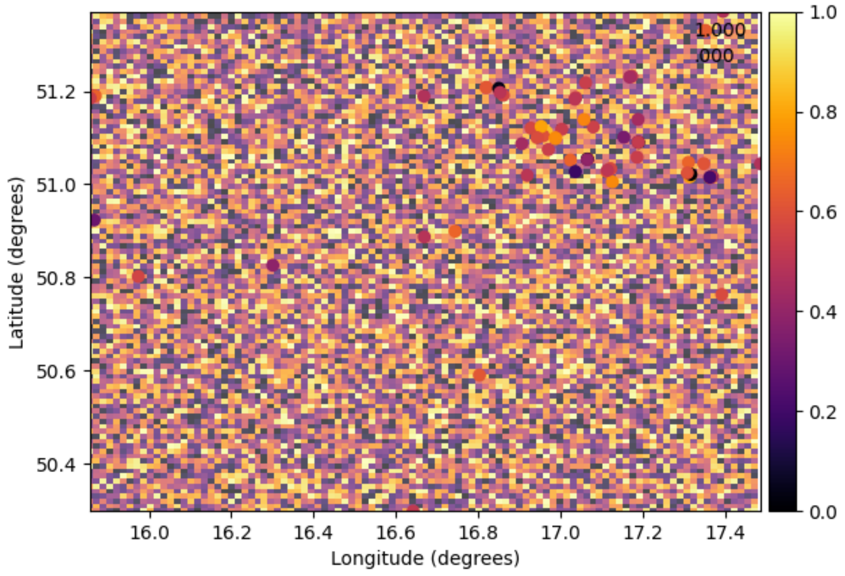

Figure 2.7 Random regression sample

2.2.4 Mean regression

The mean regression for each prediction returns an arithmetic average (equation 2.7) of all training data labels (*Y_train*). The result is independent of test data set features *Figure* 2.8.

$$y = \frac{1}{n} \cdot \sum_{i=0}^{n} Y\_train_i \tag{2.7}$$

2.2.5 Ones regression

The ones regression works pretty similar to random regression 2.2.3, but it always predicts maximum possible value in accordance with equation 2.8), *Figure* 2.9.

$$y = 1 \tag{2.8}$$

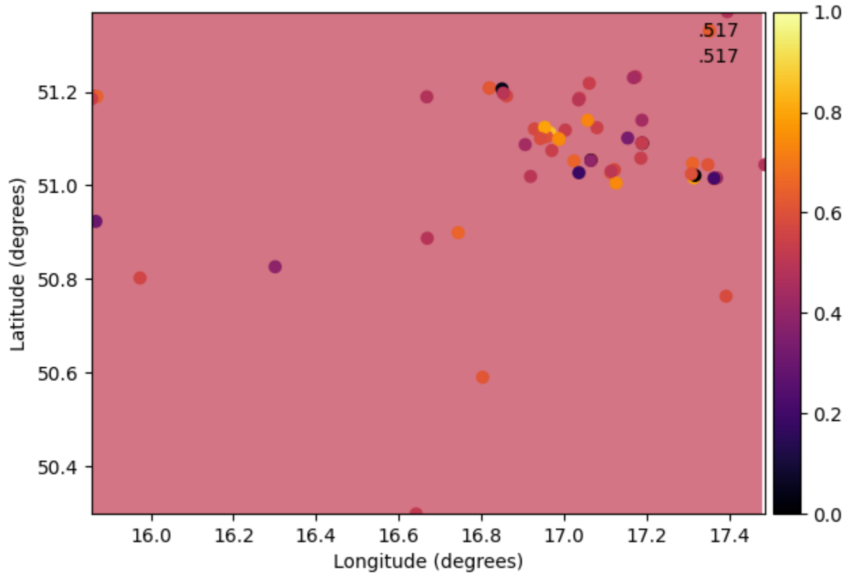

Figure 2.8 Mean regression sample

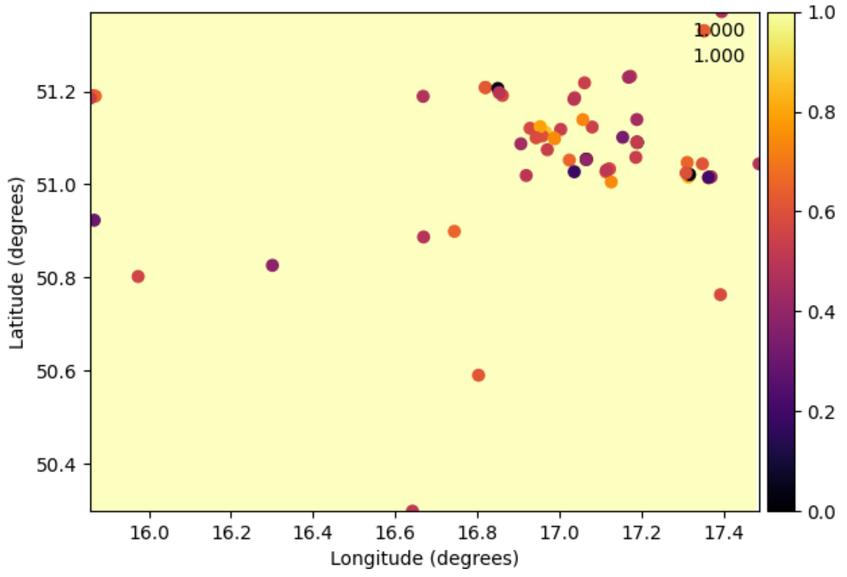

Figure 2.9 Ones regression sample

2.2.6 Linear regression

Linear regression is about adjusting the plane:

$$0 = A \cdot x + B \cdot y + C \cdot z + D \tag{2.9}$$

to the training data set. The sample adjustment is shown on the figure 2.10. The maximum and the minimum predicted value are $(0.528, 0.497)$, so it is close to average value. The same linear regression prediction, but with full range of color set was shown on the *Figure* 2.11.

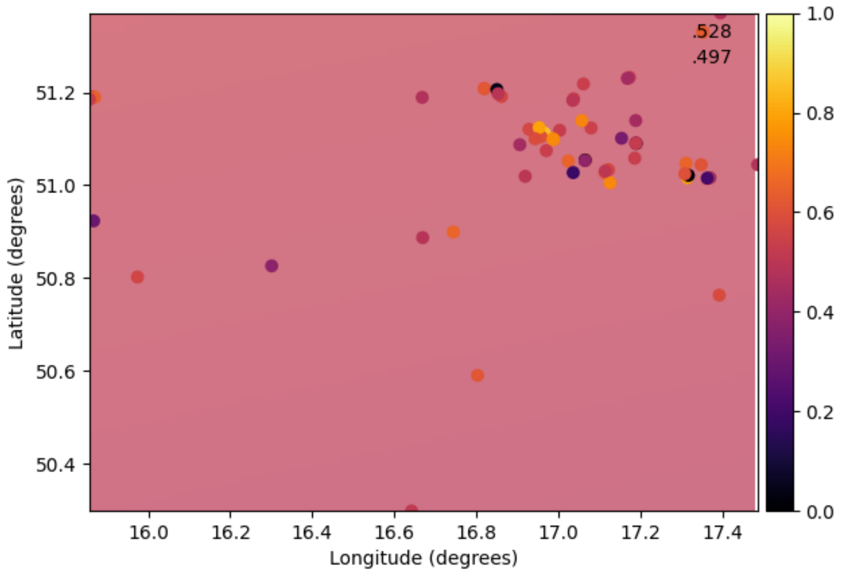

Figure 2.10 Linear regression sample

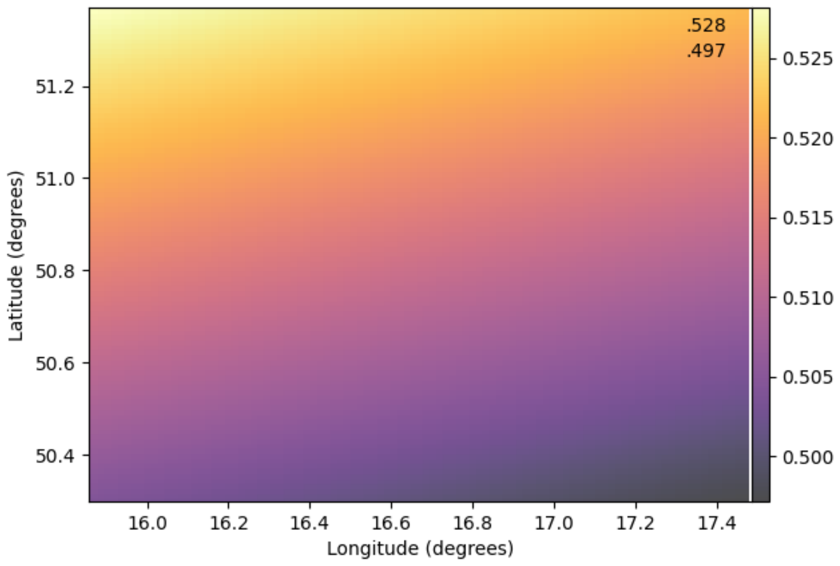

Figure 2.11 Linear regression sample without color bar normalization

It is easy to see that variation in data set are much higher than in the linear regression model, so the linear regression is probably too simple regression model.

2.2.7 Decision Tree regression

Decision tree regression is a powerful tool to deal with decision problems, it builds rectangular shape areas of constant prediction value. The areas are smaller where the density of samples is higher, and the variation of predicted values is high. The default *SK-learn* decision tree regression model† is prone to overfitting,

†https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

due to unlimited depth. (*Figure* 2.12). The output looks better if *max_depth* hyper parameter is limited *Figure* 2.13.

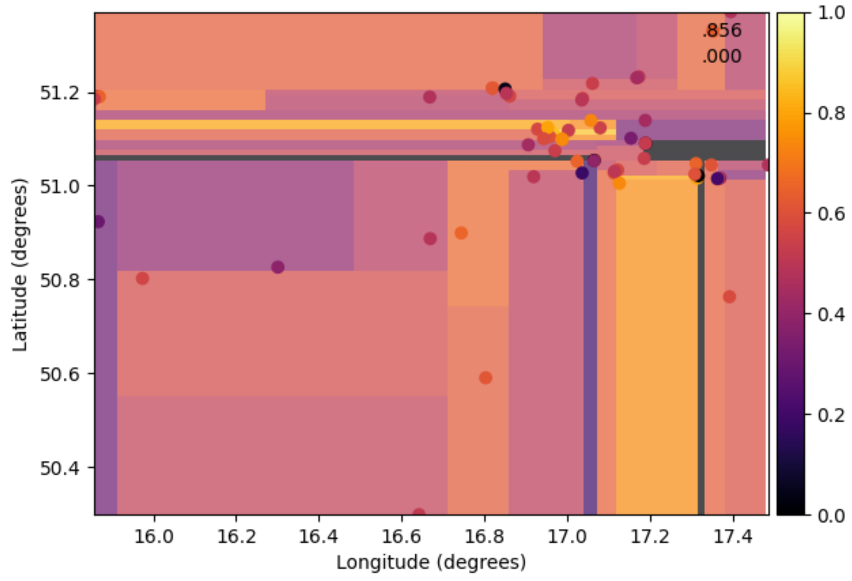

Figure 2.12 Decision tree sample

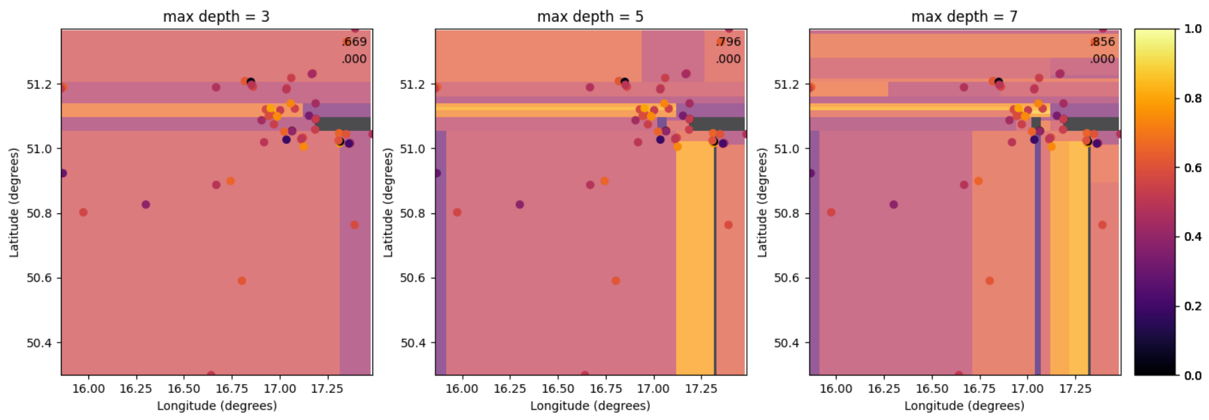

Figure 2.13 Decision tree with limited maximum depth

2.2.8 Random forest regression

Random forest is one of the ensamble regressoion [3]. It uses multiple decision tree regressoion 2.2.7. Each model is learnt on a part of training set. When model is asked to predict then each decision tree predicts prediction and random forest aggregates all individual predictions into final response. Random forest regression sample is shown on the *Figure* 2.14.

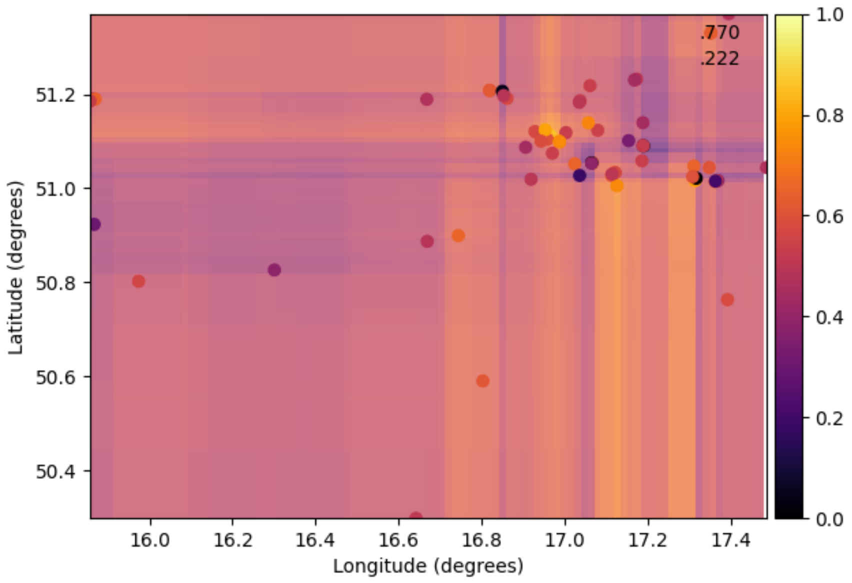

Figure 2.14 Random forest regression sample

2.2.9 2D interpolation

2D interpolation method has been taken from *SciPy* library [26]. Interpolation is a very similar process to regression but there is one significant difference. If interpolation algorithm is asked for a sample which has been used in fitting process then an interpolation model has to return exact the same value in contrast to regression which does not have to.

For purpose of this paper various methods of 2D interpolation has been used.

1. **Regular grid based interpolation** - The simplest used interpolation model is interpolation which works on 2D-grid. It builds a plane, defined by chosen kernel, between neighbours samples. Unfortunately data set which has been used in this work is not consists of regular grid, which reduces the usefulness of this method in the research figure 2.15.

2. **Spline interpolation** - Spline interpolation model create splines [25] based on learning data set. This behavior takes away the ability to extrapolate data (*Figure* 2.18) which results in incorrect interpolation in black areas. To restore this areas the learning data set was extended by added mean value on the borders in regular gaps. After fitting with extended data set result is gapless *Figure* 2.17.

3. **Radial basis functions interpolation** - A RBF interpolation [32] is an interpolation method based on weighted sum of radial functions [14]. Sample generated by two models with different kernels is shown on figure 2.18.

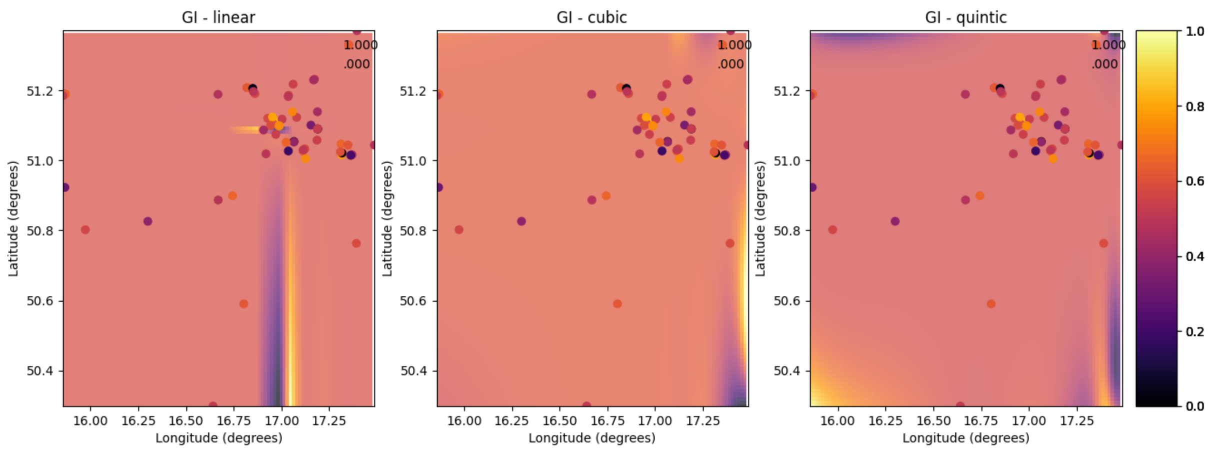

Figure 2.15 Regular grid based interpolation of different kernels

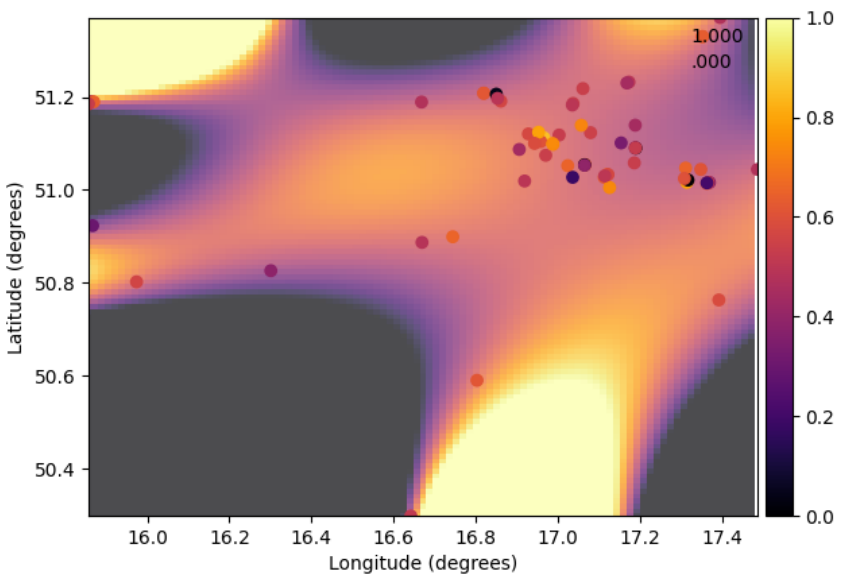

Figure 2.16 Pure *SciPy* spline interpolation sample, which show black areas

2.2.10 Nearest neighbour interpolation

Nearest neighbour interpolation (NN) is an interpolation model which basing on defined distance metric function and search for the closest training data set sample and assigns it to prediction figure 2.19. For purpose of this thesis the euclidean distance has been used (*Equation* 2.10)

$$d(p,q) = \sqrt{\sum_{i}^{n} (q_i - p_i)^2} \tag{2.10}$$

where:
$p = [p_1, p_2, \ldots, p_n], p_i \in \mathbb{R}$
$q = [q_1, q_2, \ldots, q_n], q_i \in \mathbb{R}$

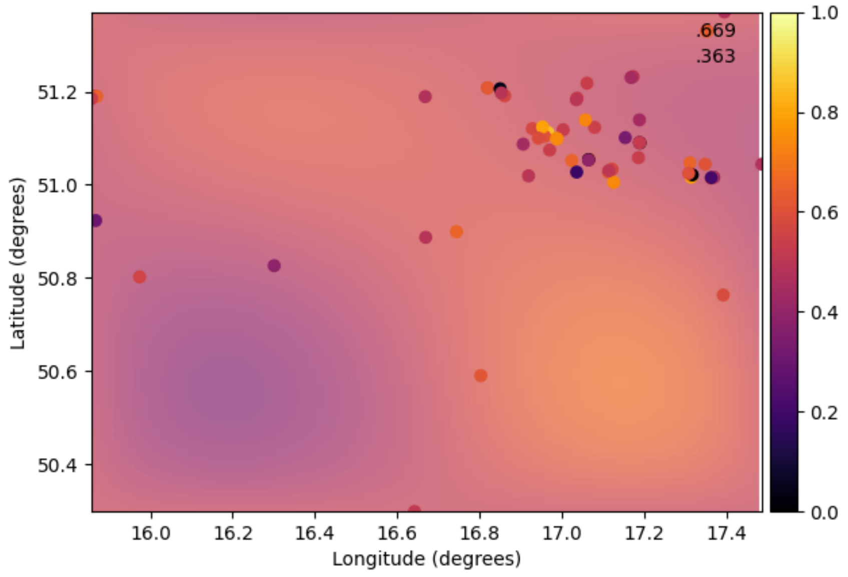

Figure 2.17 Linear interpolation with extrapolation ability

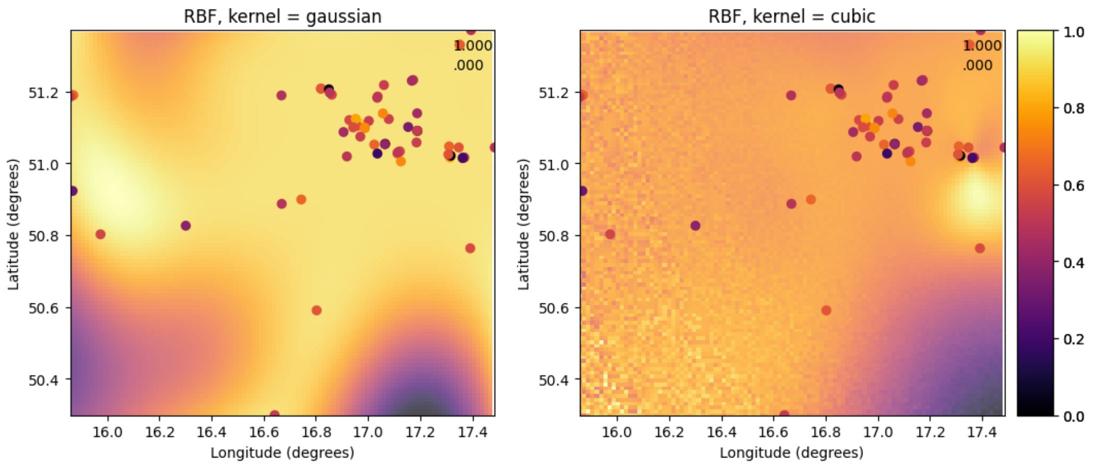

Figure 2.18 Radial basis functions interpolation

2.2.11 K - nearest neighbour regression

K - nearest neighbour (K-NN) regression is a variation on nearest neighbour interpolation 2.2.10. It works similar but add additional hyper parameter, which change the behavior. K-NN is able to assign accumulated value from more than one training data set sample.

2.2.12 Gravity regression - own method

Due to limitations of ready-to-use methods described in previous sections own method had been proposed. This method was inspired by well-known gravity equation. Prediction is a weighted average of all known data points from *Y_train* and the degree of impact (weight) is depends on the euclidean distance between data

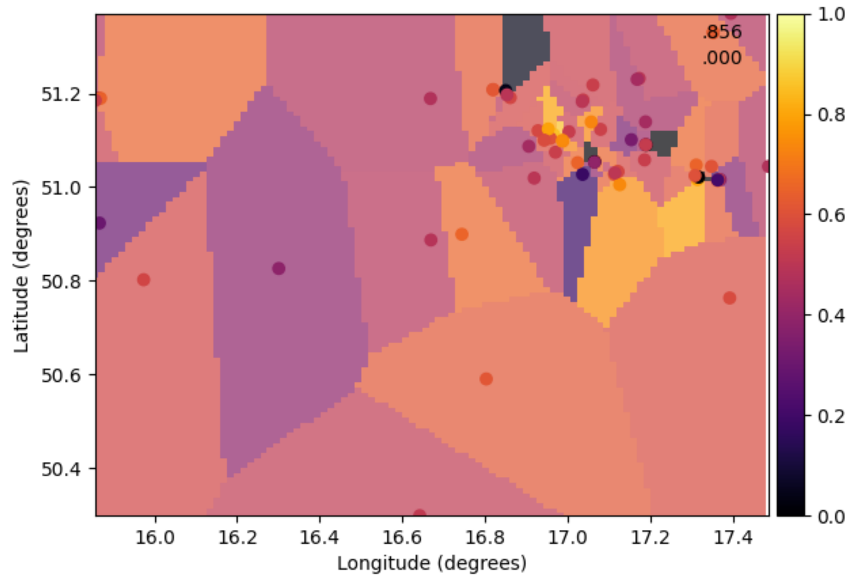

Figure 2.19 Nearest neighbour interpolation sample

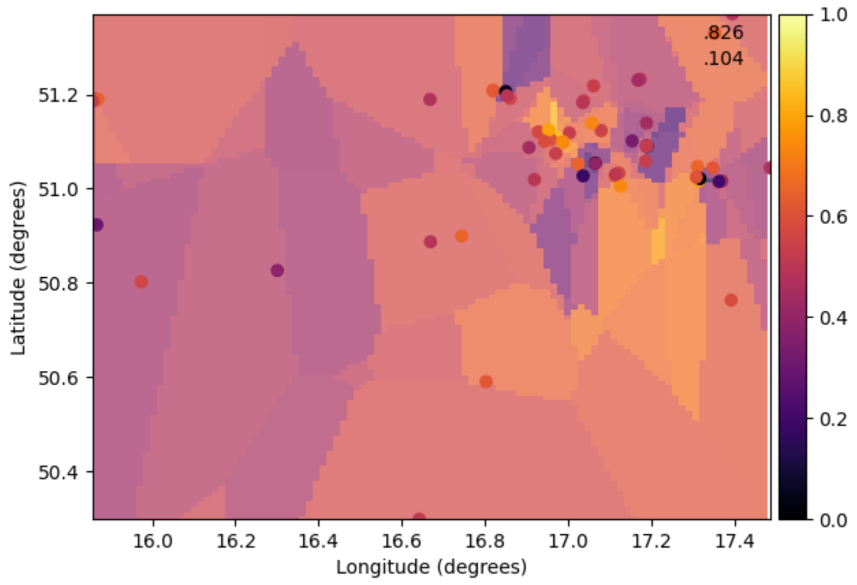

Figure 2.20 K - nearest neighbours regression sample

point and the predicted point and is given by *Equation* 2.11. Selected weight curve is called *kernel*.

$$
\begin{aligned}
&P = \frac{\sum_{i=0}^{n} Y\_train_i \cdot w_i}{\sum_{i=0}^{n} w_i},\\
&w_i = -X\_train_i + bandwidth, \text{or}\\
&w_i = e^{-(X\_train_i - X\_test)^2 \cdot bandwidth}
\end{aligned}
\tag{2.11}
$$

In case of the first kernel (figures 2.22,2.21) linear function defines relationship between weight and euclidean distance. The second available kernel is exponential function (*Figures* 2.23,2.24). There is also some limitations added to allow algorithm to work only with a local neighbourhood. If $(X\_train_i - X\_test)^2 = 0$ then w_i equals 1. If $(X\_train_i - X\_test)^2 > bandwidth$ then $w_i = 0$.

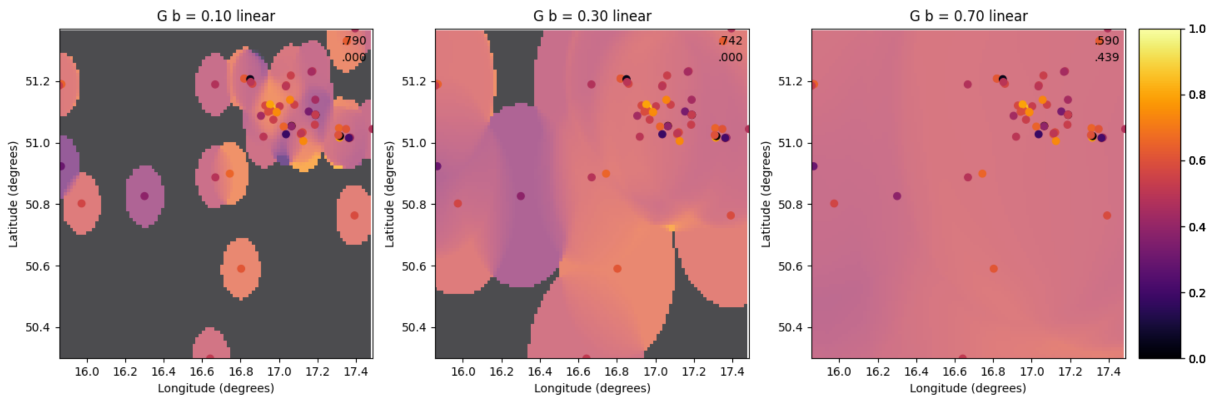

Figure 2.21 Gravity regression with linear kernel and various *bandwidth* sample

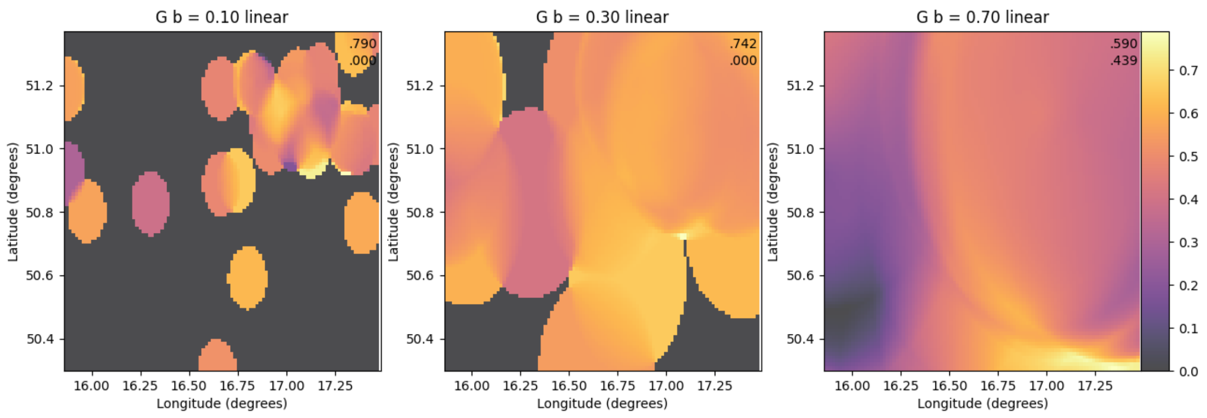

Figure 2.22 Gravity regression with linear kernel and various *bandwidth* sample shown without mesh grid color normalization

For better generalisation areas which are further away from all samples than *bandwidth*, can be replaced with *Y_train* average value (figure 2.25).

2.2.13 Training data set pre-processing

Data set used in learning process has normalized labels (Y). Due to uneven distribution of the data sources some interpolation and regression models can not work with it, or work bad. There are also some inconsistencies inside data set which

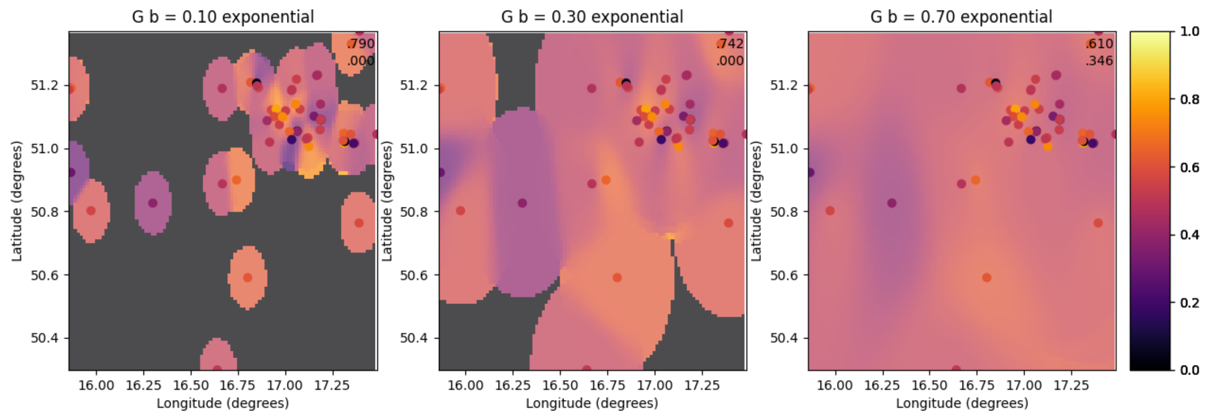

Figure 2.23 Gravity regression with exponential kernel and various *bandwidth* sample

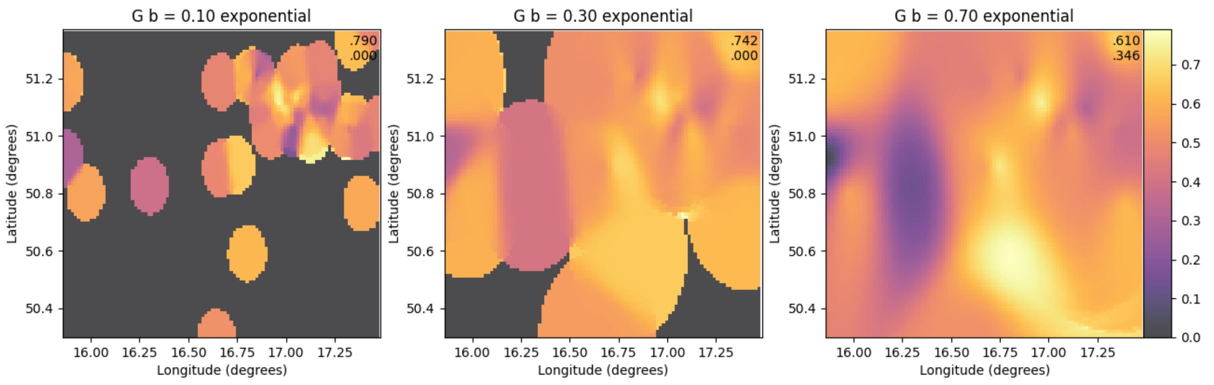

Figure 2.24 Gravity regression with exponential kernel and various *bandwidth* sample shown without mesh grid color normalization

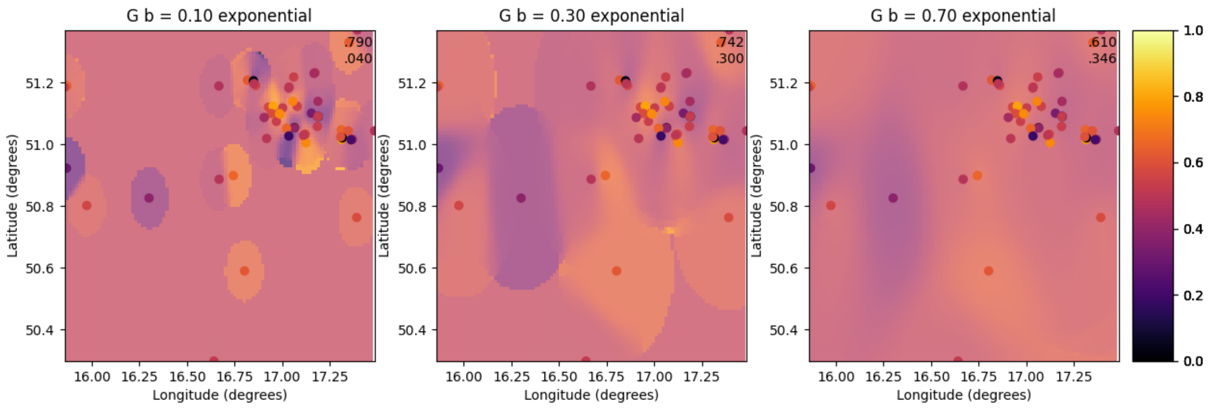

Figure 2.25 Gravity regression with exponential kernel and various *bandwidth* and mean enabled sample

have been described in section 3.2.3. Those problems had a huge influence on the models. Influence of those problems can be decreased been solved by implementing bagging and normalization in data distribution domain.

Bagging

Bootstrap aggregating (Bagging) is an ensemble machine learning technique which is used to increase stability of a model learnt on polluted data set. It is done by learning multiple models on a little bit different data set. Each data set is obtained by drawing with duplicates a certain number of samples from original data set.

In prediction process all models predict predictions and their responses are aggregated, i.e., by calculating arithmetic mean or by voting[13]. Bagging affects on decreasing variance of model predictions and makes it more stable.

Bagging has positive affect on removing incorrect data when inverter is turned off during the day time. One figures 2.26 and 2.27 this behavior was shown. On the first image where no bagging was used the black areas caused by incorrect data are clearly visible, when on the second image where, bagging regression based on 5 nearest neighbour regression, was used those areas are purple (with higher value).

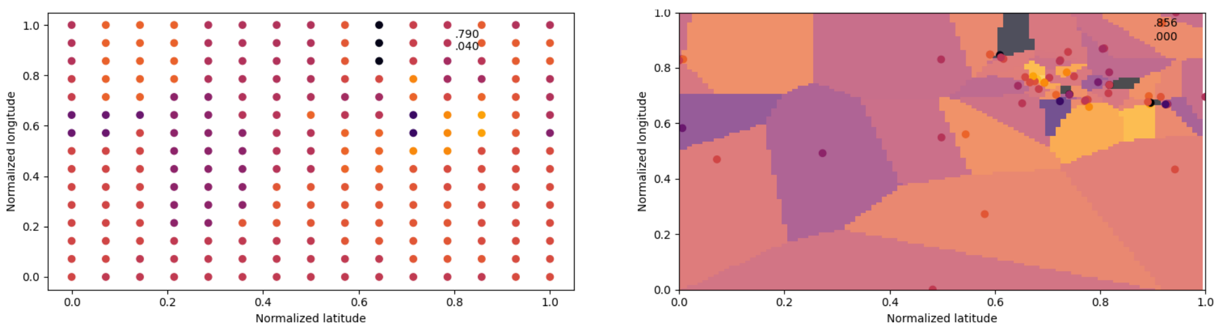

Figure 2.26 Pure nearest neighbour interpolation (mesh and samples) . Incorrect zeros data still good visible.

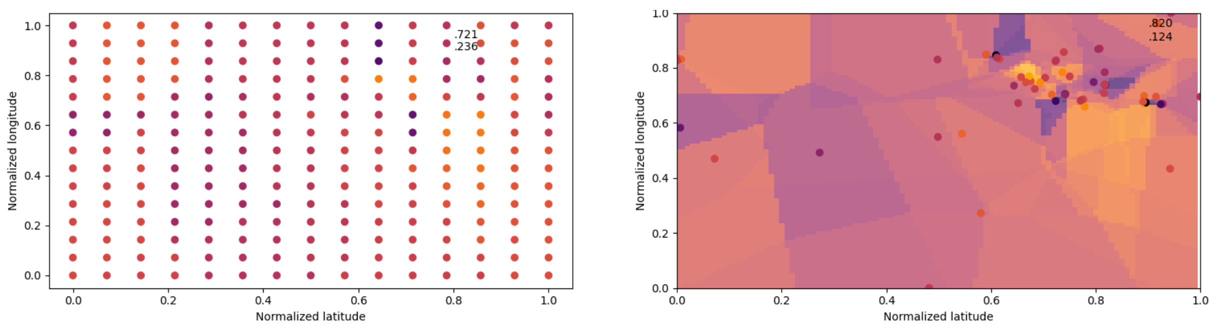

Figure 2.27 Bagging interpolation (mesh and samples) based on 5 nearest neighbour interpolations. Removed influence of incorrect zeros data in the day time.

Normalization by achieving regular grid in data set distribution domain

Generation of regular grid sampling is performed by individual or a bagging model. Sampling process is to check predictions in constant intervals. Achieved grid was shown on the *Figures* 2.26 and 2.27. On both figures on the right hand side there is an model response and on the left hand side there is a sampled regular grid where all samples are in the constant intervals between each other.

Mixing

Both methods described in *Sections* 2.2.13 and 2.2.13 has been used together in the normalization procedure. First, a model is build by bagging interpolation with 2D spline interpolation 2.2.9 or NN interpolation 2.2.10. Then obtained regressor is sampled in a regular grid of a given size. The result is a new data set which is consists of regular distributed samples.

Chapter 3

Experimental environment and research plans

This chapter aims to introduce data set, where is it taken from, programming environment and used libraries. After this introduction chapter switches into description of research plans and the research questions are stated briefly described. There are also short information what result is expected after each experiment.

3.1 Programming environment

System has been written in *Python 3.8.10* [30] with usage of various libraries. *NumPy* [10] for array manipulation and in-file storing. *Scikit-learn* [24] for machine learning algorithms, regression and model selection and *SciPy* [31] for interpolation algorithms. These libraries are powerful tools for machine learning and data analysis purposes. In this system they were used to implement both tests environment and tested models. Plots were prepared and displayed by *Matplotlib* visualization library [12].

3.2 Dataset

Database become from Remote Solar Telemetry System [23]. These system is consist of about 160 telemetry stations distributed along Poland (*Figure* 3.1 but these distribution is very uneven **??**. Most of them are located in Lower Silesia Voivodeship especially near Wrocław. But there are also some in the Silesia Voivodeship, central and north Poland.

Each telemetry station collects data from inverters build in power plants then, sends those data to the server where data are stored in database. System collects many different parameters such as: cumulative energy, active power, voltage, current, etc. Data are read from inverters in constant 2 minutes intervals. First data have been collected in April 2019.

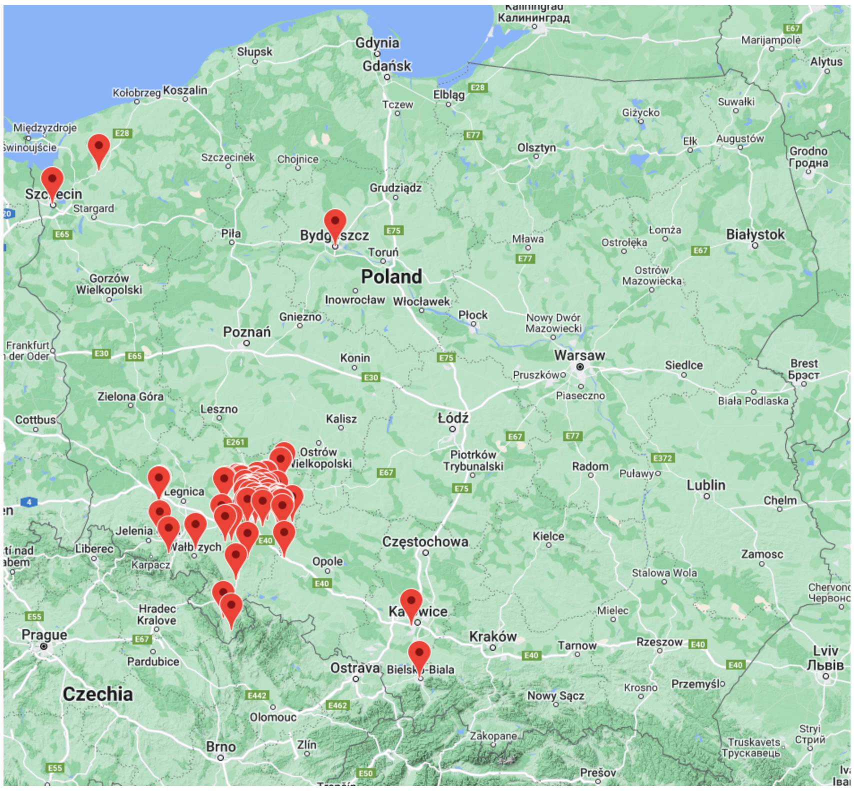

Figure 3.1 Remote Solar Telemetry System's photovoltaic power plants spread along Poland

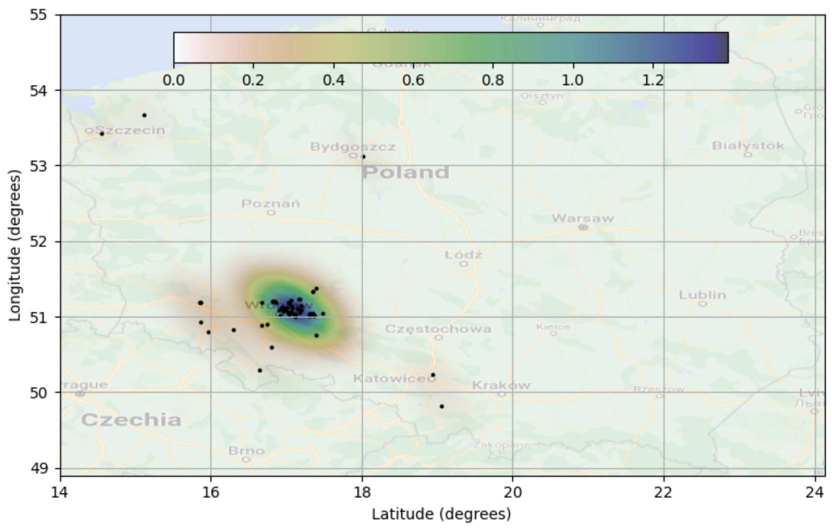

Figure 3.2 Remote Solar Telemetry System's photovoltaic power plants density

3.2.1 Dataset preparation

For research on the system under consideration dataset had to be prepared from raw data stored in database. At the beginning parameter selection was important. To estimate solar insolation the solar power plant power parameter was only one interesting parameter. Due to the different installed power on the different power plants, power parameter is inconsistent. This is because larger solar power plant produce more power than the smaller one even when the solar insolation is the same for both power plants. So the normalization was necessary. Power parameter was normalized in domain of mounted power regarding equation 3.1.

$$NP = \frac{P}{MP} \tag{3.1}$$

where (3.1):

NP	normalized power
P	photovoltaic power plant power (kW)
MP	photovoltaic power plant mounted power (kW)

Table 3.1 CSV dataset file

Elmecotechnologie	pv data			
2021-07-31T22:00:00	2021-07-31T22:00:00	600	1627768800048	...
			2021-07-31T22:00:00	...
TyniecMaly	51.019	16.919	0	...
Szczecin	53.428	14.551	0	...
Nowogard	53.673	15.118	0	...
BialeBlotoR	51.33	17.352	0	...
BialeBlotoW	51.33	17.352	0	...
...				

The second important normalization step was to normalized data in the time domain. Due to different local time, data had not been collected in exact the same time. So to remove this effect 10-minute average values have been computed and used in the dataset instead of direct values taken from inverters.

Dataset is consists of: days and nights which has been shown on the figure 3.3, rainy days, cloudy days, sunny days, winter, summer. It is a cross-section of all possible weather conditions.

3.2.2 On-disk storage

After all pre-processing operations data was exported to the *CSV* file. The beginning of the file looks like *Table* 3.1. In the top row there are company name and file description. In the second row there are information about period which the file contains and the time interval between samples. Rest cells contains the epoch

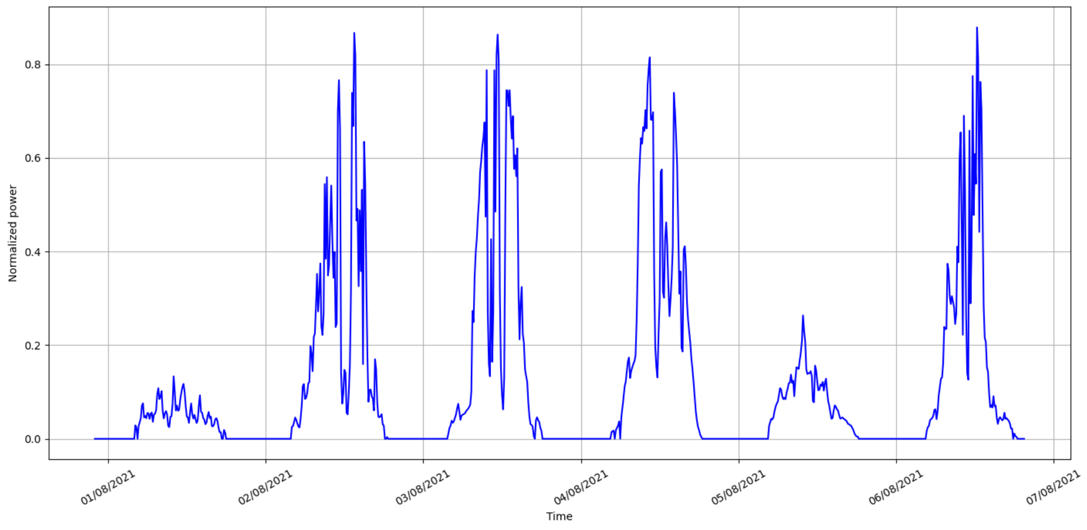

Figure 3.3 Changes in the ten-minute mean power for the Tyniec Mały installation

time timestamps [11] formatted in milliseconds. In the third row there are time-stamps in human-readable format. The rest rows are formatted as follows: the first column is installation name, then latitude, longitude and normalized production for the corresponding timestamp.

Regarding normalization power production is a decimal fraction with three decimal places (*Table* 3.2).

Table 3.2 CSV dataset file (2)

1646478600076	1646479200076	1646479800076	1646480400076
2022-03-05T11:10:00	2022-03-05T11:20:00	2022-03-05T11:30:00	2022-03-05T11:40:00
0.104	0.089	0.102	0.162
0.199	0.221	0.173	0.168
0.116	0.106	0.125	0.145
0.352	0.305	0.22	0.328
0.391	0.273	0.213	0.262

3.2.3 Dataset issues

During research some problems with dataset appeared. In this work, there was an assumption made: the power of installation is a function of solar insolation. In most cases this is true, but not always. An inconsistent was observed in dataset: sometimes there is zero production during the day. This was due to two phenomena: manually turning off inverters or over voltage in the grid. In both cases even when the sun shine on installation it did not produce energy.

3.3 Models comparison

This section aims to clearly describe evaluation protocol, which has been used in this research, testes which has been planned and the expected results.

3.3.1 Used evaluation protocol

In *Chapter* 2 evaluation protocol has been described generally, however there are some parameters which have to be defined before testes begin. The following assumptions have been made:

- Obtained results are evaluated by R^2 metric.
- Models are evaluated by 10 times 10 folds cross-validation.
- Each test is conducted on 7 datasets.
- Dataset will be filtered only to Lower-Silesia Voivodoship, where the station density is the highest.
- The obtained results will be statistically analyzed with *t-test* and significance level $\alpha = 0,05$.

3.3.2 Tests planning

At the beginning tests plans are described. It covers comparison between methods with or without normalization and bagging and with various hyper parameters. All results are achieved by averaging obtained values according assumptions. In this work the following tests were planned and conducted.

Research Question 1 *Which models are statistically inferior to random regression?*

The first test is to check how each model deal with solving static solar insulation problem. The hyper parameters are randomly set. After this step some methods which are worse than random regression, may be rejected from further testing.

Research Question 2 *Are there statistically significant differences in prediction accuracy of Decision Tree regression with various hyperparameter* max_depth *setting?*

This test focuses on comparison between various height Decision Trees. This test show how limitation of *max_depth* affect in prediction quality.

Research Question 3 *Are there statistically significant differences in prediction accuracy of Random forests regression with various hyperparameter* n_estimator *setting?*

This test focuses on comparison between various height and count of Decision Trees included in Random forest. This test show how calibration of hyperparameter affect in prediction quality. The second expected observation is the fall of the variance value witch increasing of *n_estimators*.

Research Question 4 *Is there statistically significant difference in prediction accuracy of K-NN regression and and NN interpolation?*

In this test K-NN with various K and NN interpolation are compared. The result of this test should apart of simply comparison, shows if K-NN regression is implemented correctly. K-NN for $\mathrm{K} = 1$ should works like NN interpolation.

Research Question 5 *Are there statistically significant differences in prediction accuracy of Gravity regression with various hyperparameters* kernel, bandwidth *and* mean *setting?*

This test aims to find best possible combination of kernel and bandwidth hyperparameters for prediction accuracy of Gravity regression.

Research Question 6 *Are there statistically significant differences in prediction accuracy of best methods designated during previous tests and mean regression?*

This test aims to find out if any of prepared regression model is better than simple mean regression.

Research Question 7 *Are there statistically significant differences in prediction accuracy of best methods with or without normalization?*

This is the final test. It aims to find out if best models work better or worse with normalization.

Chapter 4

Overview of the research and tests performed

In this chapter the results of performed research and tests are shown and described. Each section in this chapter covers a separate test, which answers for one of the research questions, which were posed in the previous chapter.

Each section is consists of a few elements. First, test is described more precisely than in the previous chapter. Second, tests results are shown. At the end, performed statistical analysis is shown and described.

Note: Statistical analysis is shown in two tables with some cells coloured. The size of this table corresponds to count of regression models used in the test. The first table contains *T-Test* results and the second contains *P-Value* results. A value in a cell is formed by computation of appreciate metric where the first data vector becomes from estimator in a row and the second data vector becomes from the estimator in a column. Additionally to make results of the analysis clearly visible some cells are coloured. For the *P-Value* table coloured cell means that the value inside is lower than significance level $\alpha = 0.05$. For the *T-Test* table coloured cell means that the value of this cell (*T-Test* value) is positive and corresponding *P-Value* is lower than α (corresponding cell was also marked). If cell is coloured in *T-Test* it means that regression model in a row is statistically significantly better than a regression model in a column.

4.1 Various models comparison

Which models are statistically inferior to random regression?

This test aims to check which of proposed regression models are definitely not suited for the problem under consideration. In this test various models are tested and compared with random regression. In the *Table* 4.1 results of this test was shown. As it is clearly visible the *Random* regression achieved $R^2 = -6.31 \pm 8.147$.

Only three regression model achieve worse results. They are *Ones*, *2D Regular grid based interpolation* and *2D RBF interpolation*.

Table 4.1 Comparison of various models scored by R^2 metric.

No.	Model	Hyper parameters	R^2
e1.	Random 2.2.3	—	-6.31 ± 8.147
e2.	Mean 2.2.4	—	-0.33 ± 0.589
e3.	Ones 2.2.5	—	-23.25 ± 25.696
e4.	Linear reg.2.2.6	—	-0.41 ± 0.701
e5.	Decision Tree 2.2.7	—	-2.70 ± 3.222
e6.	Decision Tree 2.2.7	max_depth = 3	-1.38 ± 2.046
e7.	Decision Tree 2.2.7	max_depth = 5	-2.26 ± 2.967
e8.	Random forest 2.2.8	—	-1.18 ± 1.516
e9.	2D inter. Grid 2.2.9	ker. = linear	-11.93 ± 14.375
e10.	2D inter. Grid 2.2.9	ker. = quintic	-14.05 ± 15.985
e11.	2D inter. Spline 2.2.9	—	-3.00 ± 4.193
e12.	2D inter. RBF 2.2.9	ker. = gaussian	-11.88 ± 14.020
e13.	NN inter. 2.2.10	—	-3.20 ± 5.052
e14.	K-NN 2.2.11	K = 3	-1.32 ± 2.163
e15.	K-NN 2.2.11	K = 5	-0.87 ± 1.580
e16.	GR 2.2.12	ker. = linear	-0.37 ± 0.635
e17.	GR 2.2.12	ker. = exp.	-0.34 ± 0.586
$\sum$	—	—	-4.99 ± 6.087

The statistical analysis of the results is shown in the *Table* 4.2. The table contains only comparison of estimator *e1* (*Random*) with each other. Positive *T-Test* values were achieved in all cases where R^2 is lower than *Random* result. And for all of these regression models α condition is met.

After this analysis we can say that *Ones*, *2D Regular grid based interpolation* and *2D RBF interpolation* are statistically significant inferior to random regression.

Table 4.2 *T-Test* and *P-Value* of Random regression model with all others models used in the first test.

T-Test

	e2	e3	e4	e5	e6	e7	e8	e9
e1	-19.366	16.613	-19.061	-10.893	-15.501	-12.357	-16.359	9.001

	e10	e11	e12	e13	e14	e15	e16	e17
e1	11.411	-9.549	9.084	-8.564	-15.645	-17.328	-19.211	-19.314

P-Value

	e2	e3	e4	e5	e6	e7	e8	e9
e1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

	e10	e11	e12	e13	e14	e15	e16	e17
e1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

4.2 Decision Tree max_depth parameterization

Are there statistically significant differences in prediction accuracy of Decision Tree regression with various hyperparameter max_depth *setting?*

Table 4.3 Comparison of *Tree* regression with various max_depth scored by R^2 metric.

No.	Model	Hyper parameters	R^2
e1.	Decision Tree 2.2.7	—	-2.88 ± 3.635
e2.	Decision Tree 2.2.7	max_depth = 1	-0.55 ± 1.000
e3.	Decision Tree 2.2.7	max_depth = 2	-0.98 ± 1.553
e4.	Decision Tree 2.2.7	max_depth = 3	-1.37 ± 2.031
e5.	Decision Tree 2.2.7	max_depth = 4	-1.77 ± 2.497
e6.	Decision Tree 2.2.7	max_depth = 5	-2.30 ± 2.983
e7.	Decision Tree 2.2.7	max_depth = 6	-2.59 ± 3.391
$\sum$	—	—	-1.78 ± 2.441

Table 4.4 *T-Test* and *P-Value* of *Tree* regression models with various *max_depth*.

T-Test

	e1	e2	e3	e4	e5	e6	e7
e1	—	-16.407	-12.734	-9.632	-6.669	-3.280	-1.545
e2	16.407	—	6.240	9.607	12.064	14.761	15.323
e3	12.734	-6.240	—	4.000	7.117	10.382	11.436
e4	9.632	-9.607	-4.000	—	3.324	6.841	8.203
e5	6.669	-12.064	-7.117	-3.324	—	3.596	5.160
e6	3.280	-14.761	-10.382	-6.841	-3.596	—	1.714
e7	1.545	-15.323	-11.436	-8.203	-5.160	-1.714	—

P-Value

	e1	e2	e3	e4	e5	e6	e7
e1	—	0.000	0.000	0.000	0.000	0.001	0.123
e2	0.000	—	0.000	0.000	0.000	0.000	0.000
e3	0.000	0.000	—	0.000	0.000	0.000	0.000
e4	0.000	0.000	0.000	—	0.001	0.000	0.000
e5	0.000	0.000	0.000	0.001	—	0.000	0.000
e6	0.001	0.000	0.000	0.000	0.000	—	0.087
e7	0.123	0.000	0.000	0.000	0.000	0.087	—

The second test aims to check if customization of *max_depth* hyperparameter affect in *Tree* regression models results. In *Table* 4.3 could be seen an relationship between R^2 metric score and *max_depth*. The higher *max_depth* the worse result is. The test shows that Tree with limited max_depth $= 1$ is statistically significantly better than others trees with higher *max_depth* (*Table* 4.4). This behavior could be caused by model overfitting for higher *max_depth* limits.

4.3 Random forest parameterization

Are there statistically significant differences in prediction accuracy of Random forests regression with various hyperparameter n_estimator setting?

Table 4.5 Comparison of *Random forest* regression with various max_depth and n_estimators scored by R^2 metric.

No.	Model	Hyper parameters	R^2
e1.	Decision Tree 2.2.7	max_depth = 1	-0.55 ± 1.000
e2.	Random forest 2.2.8	n_estimators = 25	-0.47 ± 0.758
e3.	Random forest 2.2.8	n_estimators = 50	-0.46 ± 0.757
e4.	Random forest 2.2.8	n_estimators = 100	-0.45 ± 0.741
e5.	Random forest 2.2.8	n_estimators = 200	-0.45 ± 0.734
e6.	Random forest 2.2.8	n_estimators = 300	-0.45 ± 0.728
e7.	Random forest 2.2.8	n_estimators = 400	-0.45 ± 0.732
e8.	Random forest 2.2.8	n_estimators = 800	-0.45 ± 0.740
$\sum$	—	—	-0.46 ± 0.774

In this test various *Random forest* models were tested. Regarding conclusions from 4.2 trees height was limited to 1. Each forest has been also set to a specific value of *n_estimators*. The results are presented in *Table* 4.5. Obtained R^2 scores are close to each other. The second observation is that the *variance* decreased with *n_estimators* increase in range from 25 to 300. The obtained variance is respectively $0.758, 0.757, 0.741, 0.734$ and 0.728. This observation corresponds to the theory that *Ensemble* regression method should be characterized by results with lower variance. As *Table* 4.6 shown there are estimators *e4-e8* are statistically significantly better than *e1*.

Additionally, to better visualize the declining variance with an increase in the number of estimators. In *Table* 4.7 the second test was shown. 9 models with *n_estimators* set for values from 10 to 90 were considered. Results was visualized on *Figure* 4.1. This image shows scatters of variance and a trend line with negative direction factor.

4.4 K-NN parameterization and comparison with NN interpolation

Is there statistically significant difference in prediction accuracy of K-NN regression and and NN interpolation?

Table 4.6 *T-Test* and *P-Value* of *Random forest* regression with various n_estimators.

T-Test

	e1	e2	e3	e4	e5	e6	e7	e8
e1	—	-1.631	-1.691	-2.034	-2.099	-2.121	-2.083	-2.015
e2	1.631	—	-0.069	-0.458	-0.529	-0.549	-0.508	-0.434
e3	1.691	0.069	—	-0.389	-0.459	-0.479	-0.438	-0.364
e4	2.034	0.458	0.389	—	-0.070	-0.088	-0.047	0.025
e5	2.099	0.529	0.459	0.070	—	-0.018	0.023	0.095
e6	2.121	0.549	0.479	0.088	0.018	—	0.041	0.113
e7	2.083	0.508	0.438	0.047	-0.023	-0.041	—	0.072
e8	2.015	0.434	0.364	-0.025	-0.095	-0.113	-0.072	—

P-Value

	e1	e2	e3	e4	e5	e6	e7	e8
e1	—	0.103	0.091	0.042	0.036	0.034	0.037	0.044
e2	0.103	—	0.945	0.647	0.597	0.583	0.612	0.664
e3	0.091	0.945	—	0.698	0.646	0.632	0.662	0.716
e4	0.042	0.647	0.698	—	0.944	0.930	0.962	0.980
e5	0.036	0.597	0.646	0.944	—	0.986	0.982	0.925
e6	0.034	0.583	0.632	0.930	0.986	—	0.968	0.910
e7	0.037	0.612	0.662	0.962	0.982	0.968	—	0.942
e8	0.044	0.664	0.716	0.980	0.925	0.910	0.942	—

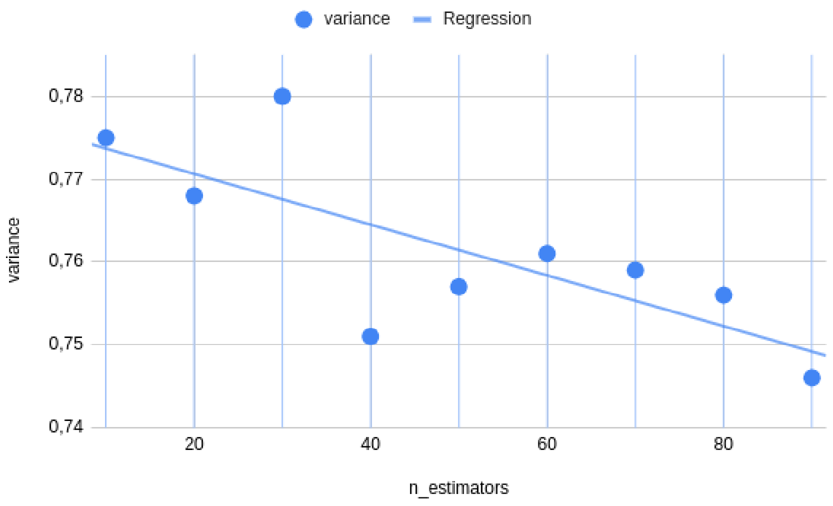

Figure 4.1 Variance of R^2 as a function of number of *n_estimators* for *Random forests* regression. Chart contains data form *Table* 4.7.

In *Table* 4.8 results of the fourth test are shown. The test covers comparison of *K-NN* regression and *NN* interpolation. *K-NN* regression has one hyperparameter which can be tunned. The following values has been chosen to be used in this test $\{1, 2, 3, 4, 5, 6, 7, 8, \infty\}$.

Table 4.7 Comparison of *Random forest* regression with various n_estimators in range from 10 to 90 scored by R^2 metric.

No.	Model	Hyper parameters	R^2
e1.	Random forest 2.2.8	n_estimators = 10	-0.48 ± 0.775
e2.	Random forest 2.2.8	n_estimators = 20	-0.47 ± 0.768
e3.	Random forest 2.2.8	n_estimators = 30	-0.47 ± 0.780
e4.	Random forest 2.2.8	n_estimators = 40	-0.46 ± 0.751
e5.	Random forest 2.2.8	n_estimators = 50	-0.46 ± 0.757
e6.	Random forest 2.2.8	n_estimators = 60	-0.47 ± 0.761
e7.	Random forest 2.2.8	n_estimators = 70	-0.46 ± 0.759
e8.	Random forest 2.2.8	n_estimators = 80	-0.45 ± 0.756
e9.	Random forest 2.2.8	n_estimators = 90	-0.45 ± 0.746
$\sum$	—	—	-0.46 ± 0.761

Table 4.8 Comparison of *K-NN* regression and *NN* interpolation with various K scored by R^2 metric.

No.	Model	Hyper parameters	R^2
e1.	NN inter. 2.2.10	—	-3.20 ± 5.052
e2.	K-NN 2.2.11	K = 1	-3.20 ± 5.052
e3.	K-NN 2.2.11	K = 2	-1.80 ± 2.288
e4.	K-NN 2.2.11	K = 3	-1.32 ± 2.163
e5.	K-NN 2.2.11	K = 4	-0.93 ± 1.432
e6.	K-NN 2.2.11	K = 5	-0.87 ± 1.580
e7.	K-NN 2.2.11	K = 6	-0.72 ± 1.178
e8.	K-NN 2.2.11	K = 7	-0.65 ± 0.914
e9.	K-NN 2.2.11	K = 8	-0.62 ± 0.924
e10.	K-NN 2.2.11	K = ∞	-0.33 ± 0.589
$\sum$	—	—	-1.36 ± 2.117

The first hypontesis to be checked is that *NN* interpolation and *K-NN* with $K = 1$ are the same models. It is proved because *T-Test* (Table 4.9) on crossing *e1* and *e2* shows a value of zero and *P-Value* of this metrics is 1.

As it is shown in the Table 4.9 model with bigger value of *K* is better than model with smaller *K*. The best results was achieved by estimator *e10*, which is *K-NN* with unlimited *K*. Additionally an interesting observation has been made. Estimator *e10* achieved similar results as *Mean* regression (*Table* 4.1) -0.33 ± 0.589.

There are statistically significant differences between *K-NN* with various *K* and NN interpolation. The bigger *K* the better model and the best model is for $K = \infty$.

Table 4.9 *T-Test* and *P-Value* of *K-NN* regression and *NN* interpolation with various K.

T-Test

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10
e1	—	0.000	-6.691	-9.058	-11.432	-11.656	-12.668	-13.134	-13.324	-14.959
e2	0.000	—	-6.691	-9.058	-11.432	-11.656	-12.668	-13.134	-13.324	-14.959
e3	6.691	6.691	—	-4.024	-8.492	-8.843	-11.117	-12.308	-12.693	-16.497
e4	9.058	9.058	4.024	—	-3.952	-4.450	-6.472	-7.518	-7.929	-11.735
e5	11.432	11.432	8.492	3.952	—	-0.783	-3.069	-4.358	-4.929	-10.371
e6	11.656	11.656	8.843	4.450	0.783	—	-2.039	-3.140	-3.676	-8.530
e7	12.668	12.668	11.117	6.472	3.069	2.039	—	-1.148	-1.810	-7.870
e8	13.134	13.134	12.308	7.518	4.358	3.140	1.148	—	-0.768	-7.960
e9	13.324	13.324	12.693	7.929	4.929	3.676	1.810	0.768	—	-6.987
e10	14.959	14.959	16.497	11.735	10.371	8.530	7.870	7.960	6.987	—

P-Value

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10
e1	—	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
e2	1.000	—	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
e3	0.000	0.000	—	0.000	0.000	0.000	0.000	0.000	0.000	0.000
e4	0.000	0.000	0.000	—	0.000	0.000	0.000	0.000	0.000	0.000
e5	0.000	0.000	0.000	0.000	—	0.434	0.002	0.000	0.000	0.000
e6	0.000	0.000	0.000	0.000	0.434	—	0.042	0.002	0.000	0.000
e7	0.000	0.000	0.000	0.000	0.002	0.042	—	0.251	0.071	0.000
e8	0.000	0.000	0.000	0.000	0.000	0.002	0.251	—	0.443	0.000
e9	0.000	0.000	0.000	0.000	0.000	0.000	0.071	0.443	—	0.000
e10	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	—

4.5 Gravity regression parameterization

Are there statistically significant differences in prediction accuracy of Gravity regression with various hyperparameters kernel, bandwidth and mean setting?

The fifth test aims to compare *Gravity* regression with various hyperparameters. Tunned hyperparameters were bandwidth, kernel and mean enabling was turned on or off.

Bandwidth is a parameter which define how far samples, regarding euclidean distance, are taken into account when prediction is computed. Kernel defines relationship between distance, value of the sample and prediction. Model uses two different kernel functions. The linear function and the exponential function are available. The last parameter is a mean enabling. If mean is enabled then if predicted sample is further from each individual sample from training set than bandwidth, the model assigns a mean of a training set. On the other hand, if it is not enabled then zero value is assigned. In *Table* 4.10 results of test of various *bandwidth*

Table 4.10 Comparison of *Gravity* regression with various *bandwidth* with or without mean enabled and linear kernel.

No.	Model	Hyper parameters	R^2
e1.	GR 2.2.12	b = 0.1	-2.14 ± 5.437
e2.	GR 2.2.12	b = 0.5	-0.37 ± 0.635
e3.	GR 2.2.12	b = 1	-0.38 ± 0.622
e4.	GR 2.2.12	b = 2	-0.33 ± 0.590
e5.	GR 2.2.12	b = 5	-0.33 ± 0.589
e6.	GR 2.2.12	mean = en,b = 0.1	-0.67 ± 0.878
e7.	GR 2.2.12	mean = en,b = 0.5	-0.37 ± 0.635
e8.	GR 2.2.12	mean = en,b = 1	-0.38 ± 0.622
e9.	GR 2.2.12	mean = en,b = 2	-0.33 ± 0.590
e10.	GR 2.2.12	mean = en,b = 5	-0.33 ± 0.589
$\sum$	—	—	-0.56 ± 1.119

Table 4.11 *T-Test* and *P-Value* of *Gravity* regression with various *bandwidth* with or without mean enabled and linear kernel.

T-Test

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10
e1	—	-8.555	-8.533	-8.748	-8.774	-7.063	-8.555	-8.533	-8.748	-8.774
e2	8.555	—	0.149	-1.172	-1.333	7.315	0.000	0.149	-1.172	-1.333
e3	8.533	-0.149	—	-1.339	-1.501	7.241	-0.149	0.000	-1.339	-1.501
e4	8.748	1.172	1.339	—	-0.166	8.452	1.172	1.339	0.000	-0.166
e5	8.774	1.333	1.501	0.166	—	8.587	1.333	1.501	0.166	0.000
e6	7.063	-7.315	-7.241	-8.452	-8.587	—	-7.315	-7.241	-8.452	-8.587
e7	8.555	0.000	0.149	-1.172	-1.333	7.315	—	0.149	-1.172	-1.333
e8	8.533	-0.149	0.000	-1.339	-1.501	7.241	-0.149	—	-1.339	-1.501
e9	8.748	1.172	1.339	0.000	-0.166	8.452	1.172	1.339	—	-0.166
e10	8.774	1.333	1.501	0.166	0.000	8.587	1.333	1.501	0.166	—

P-Value

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10
e1	—	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
e2	0.000	—	0.882	0.241	0.183	0.000	1.000	0.882	0.241	0.183
e3	0.000	0.882	—	0.181	0.134	0.000	0.882	1.000	0.181	0.134
e4	0.000	0.241	0.181	—	0.868	0.000	0.241	0.181	1.000	0.868
e5	0.000	0.183	0.134	0.868	—	0.000	0.183	0.134	0.868	1.000
e6	0.000	0.000	0.000	0.000	0.000	—	0.000	0.000	0.000	0.000
e7	0.000	1.000	0.882	0.241	0.183	0.000	—	0.882	0.241	0.183
e8	0.000	0.882	1.000	0.181	0.134	0.000	0.882	—	0.181	0.134
e9	0.000	0.241	0.181	1.000	0.868	0.000	0.241	0.181	—	0.868
e10	0.000	0.183	0.134	0.868	1.000	0.000	0.183	0.134	0.868	—

with or without mean enabled and linear kernel are shown. Bandwidth was set as $0.1, 0.5, 1, 2$ or 5. All bandwidth values was tested with mean enabled or disabled. The worst result was achieved by estimator *e1* it was -2.14 ± 5.437 far above the average value of the measurement results -0.56. As expected after mean enabled estimator *e6* result improved to -0.67 ± 0.878. For others *bandwidth* enabling or disabling mean make no differences in prediction. The analysis of this test is shown in *Table* 4.11. Estimators with the lowest bandwidth *e1* and *e6* are statistically inferior to all others models. The interesting observation is *T-Test* for estimator pairs: (*e2, e7*), (*e3, e8*), (*e4, e9*) and (*e5, e10*) assigns 0. It points that $\mathrm{bandwidth} \geqslant 0.5$ is big enough to avoid places where no training sample belongs.

Table 4.12 Comparison of *Gravity* regression with various *bandwidth* with or without mean enabled and exponential kernel.

No.	Model	Hyper parameters	R^2
e1.	GR 2.2.12	$b = 0.1$	-2.90 ± 6.190
e2.	GR 2.2.12	$b = 0.5$	-0.60 ± 0.791
e3.	GR 2.2.12	$b = 1$	-0.43 ± 0.652
e4.	GR 2.2.12	$b = 2$	-0.35 ± 0.593
e5.	GR 2.2.12	$b = 5$	-0.34 ± 0.587
e6.	GR 2.2.12	$\mathrm{mean} = \mathrm{en}, b = 0.1$	-1.43 ± 1.841
e7.	GR 2.2.12	$\mathrm{mean} = \mathrm{en}, b = 0.5$	-0.60 ± 0.791
e8.	GR 2.2.12	$\mathrm{mean} = \mathrm{en}, b = 1$	-0.43 ± 0.652
e9.	GR 2.2.12	$\mathrm{mean} = \mathrm{en}, b = 2$	-0.35 ± 0.593
e10.	GR 2.2.12	$\mathrm{mean} = \mathrm{en}, b = 5$	-0.34 ± 0.587
$\sum$	—	—	-0.78 ± 1.328

The second test was performed for the same set of hyperparametes apart from kernel which has been set to exponential function. Results are shown in *Table* 4.14 and analysis is shown in *Table* 4.15. In this case results are pretty similar to those from previous test. Estimator *e1* gave the worst R^2 score. Estimator *e6* (with mean enabled) gave statistically significant better results than *e1*. For others estimators enabling or disabling mean made no differences.

The third test aims to compare various kernels with different bandwidth. Results of this test are shown in *Table* 4.14 and analysis is shown in *Table* 4.15. Apart from the same conclusions like in to previous cases, that the smaller *bandwidth* is worse than the bigger, this result shown that there is no statistically significant difference between two kernel functions. We can pose only that for smaller $\mathrm{bandwidth} \leqslant 1$ linear kernel works better than exponential (estimators *e2-e5* are better than *e6, e7* and *e4, e5* are also better than *e8*). The same behavior is observed for exponential kernel (estimators *e8-e10* are better than *e6, e7* and *e9, 10* are also better than *e8*).

Conclusions from research question 5 state that, mean enabling has ability to improve prediction quality but only for small *bandwidths* (in tests it was 0.1) for bigger bandwidth it makes no differences. Tests shows also not statistically important differences in prediction for various kernel function. Only one hyperparameter which has statistically significant affect on model is *bandwidth*. The bigger bandwidth the better prediction.

Table 4.13 *T-Test* and *P-Value* of *Gravity* regression with various *bandwidth* with or without mean enabled and exponential kernel.

T-Test

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10
e1	—	-9.716	-10.474	-10.816	-10.875	-6.024	-9.716	-10.474	-10.816	-10.875
e2	9.716	—	-4.445	-6.700	-7.090	10.845	0.000	-4.445	-6.700	-7.090
e3	10.474	4.445	—	-2.342	-2.764	13.459	4.445	0.000	-2.342	-2.764
e4	10.816	6.700	2.342	—	-0.434	14.660	6.700	2.342	0.000	-0.434
e5	10.875	7.090	2.764	0.434	—	14.860	7.090	2.764	0.434	0.000
e6	6.024	-10.845	-13.459	-14.660	-14.860	—	-10.845	-13.459	-14.660	-14.860
e7	9.716	0.000	-4.445	-6.700	-7.090	10.845	—	-4.445	-6.700	-7.090
e8	10.474	4.445	0.000	-2.342	-2.764	13.459	4.445	—	-2.342	-2.764
e9	10.816	6.700	2.342	0.000	-0.434	14.660	6.700	2.342	—	-0.434
e10	10.875	7.090	2.764	0.434	0.000	14.860	7.090	2.764	0.434	—

P-Value

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10
e1	—	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
e2	0.000	—	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000
e3	0.000	0.000	—	0.019	0.006	0.000	0.000	1.000	0.019	0.006
e4	0.000	0.000	0.019	—	0.665	0.000	0.000	0.019	1.000	0.665
e5	0.000	0.000	0.006	0.665	—	0.000	0.000	0.006	0.665	1.000
e6	0.000	0.000	0.000	0.000	0.000	—	0.000	0.000	0.000	0.000
e7	0.000	1.000	0.000	0.000	0.000	0.000	—	0.000	0.000	0.000
e8	0.000	0.000	1.000	0.019	0.006	0.000	0.000	—	0.019	0.006
e9	0.000	0.000	0.019	1.000	0.665	0.000	0.000	0.019	—	0.665
e10	0.000	0.000	0.006	0.665	1.000	0.000	0.000	0.006	0.665	—

Table 4.14 Comparison of *Gravity* regression with various *bandwidth* with or without mean enabled and exponential kernel.

No.	Model	Hyper parameters	R^2
e1.	GR 2.2.12	b = 0.1, kernel = linear	-0.67 ± 0.878
e2.	GR 2.2.12	b = 0.5, kernel = linear	-0.37 ± 0.635
e3.	GR 2.2.12	b = 1, kernel = linear	-0.38 ± 0.622
e4.	GR 2.2.12	b = 2, kernel = linear	-0.33 ± 0.590
e5.	GR 2.2.12	b = 5, kernel = linear	-0.33 ± 0.589
e6.	GR 2.2.12	b = 0.1, kernel = exp.	-1.43 ± 1.841
e7.	GR 2.2.12	b = 0.5, kernel = exp.	-0.60 ± 0.791
e8.	GR 2.2.12	b = 1, kernel = exp.	-0.43 ± 0.652
e9.	GR 2.2.12	b = 2, kernel = exp.	-0.35 ± 0.593
e10.	GR 2.2.12	b = 5, kernel = exp.	-0.34 ± 0.587
$\sum$	—	—	-0.52 ± 0.778

Table 4.15 *T-Test* and *P-Value* of *Gravity* regression with various *bandwidth* and *kernel*.

T-Test

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10
e1	—	-7.315	-7.241	-8.452	-8.587	9.781	-1.505	-5.792	-7.929	-8.295
e2	7.315	—	0.149	-1.172	-1.333	14.316	6.062	1.747	-0.545	-0.966
e3	7.241	-0.149	—	-1.339	-1.501	14.277	5.977	1.617	-0.705	-1.131
e4	8.452	1.172	1.339	—	-0.166	14.946	7.260	2.963	0.649	0.217
e5	8.587	1.333	1.501	0.166	—	15.020	7.405	3.122	0.815	0.384
e6	-9.781	-14.316	-14.277	-14.946	-15.020	—	-10.845	-13.459	-14.660	-14.860
e7	1.505	-6.062	-5.977	-7.260	-7.405	10.845	—	-4.445	-6.700	-7.090
e8	5.792	-1.747	-1.617	-2.963	-3.122	13.459	4.445	—	-2.342	-2.764
e9	7.929	0.545	0.705	-0.649	-0.815	14.660	6.700	2.342	—	-0.434
e10	8.295	0.966	1.131	-0.217	-0.384	14.860	7.090	2.764	0.434	—

P-Value

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10
e1	—	0.000	0.000	0.000	0.000	0.000	0.133	0.000	0.000	0.000
e2	0.000	—	0.882	0.241	0.183	0.000	0.000	0.081	0.586	0.334
e3	0.000	0.882	—	0.181	0.134	0.000	0.000	0.106	0.481	0.258
e4	0.000	0.241	0.181	—	0.868	0.000	0.000	0.003	0.517	0.828
e5	0.000	0.183	0.134	0.868	—	0.000	0.000	0.002	0.415	0.701
e6	0.000	0.000	0.000	0.000	0.000	—	0.000	0.000	0.000	0.000
e7	0.133	0.000	0.000	0.000	0.000	0.000	—	0.000	0.000	0.000
e8	0.000	0.081	0.106	0.003	0.002	0.000	0.000	—	0.019	0.006
e9	0.000	0.586	0.481	0.517	0.415	0.000	0.000	0.019	—	0.665
e10	0.000	0.334	0.258	0.828	0.701	0.000	0.000	0.006	0.665	—

4.6 Comparison of selected regression methods and mean regression

Are there statistically significant differences in prediction accuracy of best methods designated during previous tests and mean regression?

The goal of this test is to find out which model is the best suited for solving the problem under consideration. In previous tests the following models were found better than the others.

- *Random forest* regression $\mathtt{max\_depth} = 1$ and $\mathtt{n\_estimators} = 100$.
- *K-NN* regression $K = \infty$.
- *Gravity* regression $\mathtt{kernel} = \mathtt{linear}$, $\mathtt{bandwidth} = 5$ and mean enabled.
- *Gravity* regression $\mathtt{kernel} = \mathtt{exponential}$, $\mathtt{bandwidth} = 5$ and mean enabled.

Apart from those models also *Mean* regression is taking into account in this test. The test results are shown in *Table* 4.16 and analysis is presented in *Table* 4.17. The results shown that all models outperformed *Random forest*. Apart from, there are no statistically significant differences between the proposed models. Additionally the idea that *K-NN* with $K = \infty$ and *Mean* regression are the same model found the evidence here. In *Table* 4.17 on the crossing *e1* and *e3 T-Test* is 0. It means both series (predictions made by *Mean* and *K-NN*) are the same.

Table 4.16 Comparison of *Mean*, *Random forest*, *K-NN*, and *Gravity* regression with selected hyperparameters.

No.	Model	Hyper parameters	R^2
e1.	Mean 2.2.4	—	-0.33 ± 0.589
e2.	Random forest 2.2.8	n_estimators = 100	-0.45 ± 0.741
e3.	K-NN 2.2.11	$K = \infty$	-0.33 ± 0.589
e4.	GR 2.2.12	b = 5, kernel = exp.	-0.34 ± 0.587
e5.	GR 2.2.12	b = 5, kernel = linear	-0.33 ± 0.589
$\sum$	—	—	-0.35 ± 0.619

Table 4.17 *T-Test* and *P-Value* of *Mean*, *Random forest*, *K-NN*, and *Gravity* regression with selected hyperparameters.

T-Test

	e1	e2	e3	e4	e5
e1	—	3.440	0.000	0.443	0.059
e2	-3.440	—	-3.440	-3.056	-3.388
e3	0.000	3.440	—	0.443	0.059
e4	-0.443	3.056	-0.443	—	-0.384
e5	-0.059	3.388	-0.059	0.384	—

P-Value

	e1	e2	e3	e4	e5
e1	—	0.001	1.000	0.658	0.953
e2	0.001	—	0.001	0.002	0.001
e3	1.000	0.001	—	0.658	0.953
e4	0.658	0.002	0.658	—	0.701
e5	0.953	0.001	0.953	0.701	—

4.7 Normalization affect on prediction accuracy

Are there statistically significant differences in prediction accuracy of best methods with or without normalization?

The purpose of this test was to show if normalization could increase prediction quality of the bests regression models. Estimators *e1, e3, e5, e7, e9* are the same estimators like they were in the previous test. Estimators *e2, e4, e6, e8, e10* are the same estimators but learnt on normalized dataset.

Table 4.18 Comparison of best models with or without normalization.

No.	Model	Hyper parameters	Norm.	R^2
e1.	Mean 2.2.4	—	N	-0.33 ± 0.589
e2.	Mean 2.2.4	—	Y	-0.39 ± 0.856
e3.	Random forest 2.2.8	n_estimators = 100	N	-0.45 ± 0.741
e4.	Random forest 2.2.8	n_estimators = 100	Y	-0.46 ± 0.998
e5.	K-NN 2.2.11	$K = \infty$	N	-0.33 ± 0.589
e6.	K-NN 2.2.11	$K = \infty$	Y	-0.39 ± 0.856
e7.	GR 2.2.12	b = 5, kernel = exp.	N	-0.34 ± 0.587
e8.	GR 2.2.12	b = 5, kernel = exp.	Y	-0.37 ± 0.835
e9.	GR 2.2.12	b = 5, kernel = linear	N	-0.33 ± 0.589
e10.	GR 2.2.12	b = 5, kernel = linear	Y	-0.39 ± 0.854
$\sum$	—	—	—	-0.38 ± 0.749

Table 4.19 *T-Test* and *P-Value* of *Mean, Random forest, K-NN,* and *Gravity* regression with selected hyperparameters.

T-Test

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10
e1	—	3.440	0.000	0.443	0.059	1.370	3.046	1.370	0.985	1.323
e2	-3.440	—	-3.440	-3.056	-3.388	-1.827	0.065	-1.827	-2.226	-1.874
e3	0.000	3.440	—	0.443	0.059	1.370	3.046	1.370	0.985	1.323
e4	-0.443	3.056	-0.443	—	-0.384	0.988	2.713	0.988	0.596	0.941
e5	-0.059	3.388	-0.059	0.384	—	1.319	3.002	1.319	0.933	1.272
e6	-1.370	1.827	-1.370	-0.988	-1.319	—	1.687	0.000	-0.366	-0.044
e7	-3.046	-0.065	-3.046	-2.713	-3.002	-1.687	—	-1.687	-2.036	-1.729
e8	-1.370	1.827	-1.370	-0.988	-1.319	0.000	1.687	—	-0.366	-0.044
e9	-0.985	2.226	-0.985	-0.596	-0.933	0.366	2.036	0.366	—	0.322
e10	-1.323	1.874	-1.323	-0.941	-1.272	0.044	1.729	0.044	-0.322	—

P-Value

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10
e1	—	0.001	1.000	0.658	0.953	0.171	0.002	0.171	0.325	0.186
e2	0.001	—	0.001	0.002	0.001	0.068	0.948	0.068	0.026	0.061
e3	1.000	0.001	—	0.658	0.953	0.171	0.002	0.171	0.325	0.186
e4	0.658	0.002	0.658	—	0.701	0.323	0.007	0.323	0.551	0.347
e5	0.953	0.001	0.953	0.701	—	0.187	0.003	0.187	0.351	0.204
e6	0.171	0.068	0.171	0.323	0.187	—	0.092	1.000	0.714	0.965
e7	0.002	0.948	0.002	0.007	0.003	0.092	—	0.092	0.042	0.084
e8	0.171	0.068	0.171	0.323	0.187	1.000	0.092	—	0.714	0.965
e9	0.325	0.026	0.325	0.551	0.351	0.714	0.042	0.714	—	0.748
e10	0.186	0.061	0.186	0.347	0.204	0.965	0.084	0.965	0.748	—

Results of this test are shown in *Table* 4.18 and analysis is shown in *Table* 4.19. All models with normalization underperformed the corresponding models without normalization. *T-test* was computed for all estimators pairs although it may be computed only for pairs of the same estimator with or without normalization. Full computation gives a fuller view on the whole situation. Analysis proved that there is no statistically significant differences in selected regression methods with or without normalization.

Chapter 5

Summary, conclusions and directions for further research

The research conducted in purpose of this work focused on preparing and testing a methods which has ability to solve or help in solving incoming serious problem that will affect utility grid and *Renewable Energy Sources* market. The grid instability caused by *RES* which production we cannot control will affect on a whole market. In order to avoid it we have to invent tools which are able to forecast power produced by *RES*. In this thesis an approach to prepare such a system was proposed and tested. The research problem was called prediction of solar insulation based on photovoltaic power plant monitoring system data.

The proposed approach was based on machine learning regression and interpolation prediction models learnt by supervised learning method. Data was taken from photovoltaic solar power plan monitoring system. Those data set was consist of location of power plan and a series of normalized power production.

Different well-known machine learning models such as: Random regression, Mean regression, Linear regression, Decision Tree, Random forests, 2D regular grid interpolation, Spline interpolation, RBF interpolation, Nearest neighbour, K-Nearest neighbours have been used in this work and one own method has been proposed. Proposed methods have been tested and statistically compared using *T-Test* and *P-Value* with significance level $\alpha = 0.05$. Models were evaluated using R^2 metric. The conducted tests allowed to select the best regression methods. The best achieved result was -0.33 ± 0.589 R^2 and it was obtained by *Mean* regression. The very similar result was obtained on evaluation of *Gravity* regression with $\mathtt{bandwidth} = 1$. Unfortunately obtained R^2 metrics are not satisfying.

The proposed approach gives rather disappointing result. Proposed models have not acquired ability to predict solar insulation. There are many possible reasons why it was happen. Proposed model takes into account only current production of photovoltaic power plants, but the weather which is a main factor influencing on a solar insulation is changing in the time. The clouds are moving over the sky, humidity and temperature change, grid conditions and all other factors affect in production. All this features makes problem much more complex.

Further work should be concentrated on extension the data set by information regarding weather, temperature, weather forecast, the time of the day, sun elevation angle and all other parameters related to this subject. The further researchers should also consider changing point of view on the problem. Change it from the static problem to dynamic and apply stream learning methods to deal with it. It looks very promising because model may be able to learn specific conditions where photovoltaic power plant works, i.e., in considered case pure production was normalized regard to mounted power, but the case is more complicated. East-west installation needs much more solar insulation than south to produce the same amount of power (of course if they mounted power are the same). Model which consider a whole situation and know the history of production could be much more powerful and would be able to compensate more noise factors.

Bibliography

[1] Boualem Bendib, Hocine Belmili, and Fateh Krim. A survey of the most used mppt methods: Conventional and advanced algorithms applied for photovoltaic systems. *Renewable and Sustainable Energy Reviews*, 45:637–648, 2015.

[2] European Union Commision. European union long term climate strategy. https://ec.europa.eu/clima/eu-action/climate-strategies-targets/2050-long-term-strategy˙en, 2022.

[3] Thomas G Dietterich et al. Ensemble learning. *The handbook of brain theory and neural networks*, 2(1):110–125, 2002.

[4] Polskie Sieci Elektroenergetyczne. Amount of installed power by different energy sources in years 1960-2021. https://www.pse.pl/dane-systemowe/funkcjonowanie-kse/raporty-roczne-z-funkcjonowania-kse-za-rok/raporty-za-rok-2021#r1˙3, 2022.

[5] Polskie Sieci Elektroenergetyczne. Energy amount by different energy sources in 2021. https://www.pse.pl/dane-systemowe/funkcjonowanie-kse/raporty-roczne-z-funkcjonowania-kse-za-rok/raporty-za-rok-2021#r6˙2, 2022.

[6] Polskie Sieci Elektroenergetyczne. Shere of produced energy by different energy sources in years 1960-2021. https://www.pse.pl/dane-systemowe/funkcjonowanie-kse/raporty-roczne-z-funkcjonowania-kse-za-rok/raporty-za-rok-2021#r6˙3, 2022.

[7] Georgios A. Florides and Paul Christodoulides. Global warming and carbon dioxide through sciences. *Environment International*, 35(2):390–401, 2009.

[8] National Fund for Environmental Protection and Water Management. National fund for environmental protection and water management. https://www.gov.pl/web/nfosigw, 2022.

[9] National Fund for Environmental Protection and Water Management. Program priorytetowy mój prąd. https://mojprad.gov.pl/, 2022.

[10] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy,

Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[11] Elliott Hauser. Unix time, utc, and datetime: Jussivity, prolepsis, and incorrigibility in modern timekeeping. *Proceedings of the Association for Information Science and Technology*, 55(1):161–170, 2018.

[12] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[13] Breiman Leo. Machine learning. *Kluwer Academic Publishers*, pages 123–140, 1996.

[14] Zuzana Majdisova and Vaclav Skala. Radial basis function approximations: comparison and applications. *Applied Mathematical Modelling*, 51:728–743, 2017.

[15] Ministry of Climate and Environment. New rules for prosumers' billing from 2022. https://www.gov.pl/attachment/47e43da4-8258-4844-b158-77f3f6b607b8, 2021.

[16] K.N. Nwaigwe, P. Mutabilwa, and E. Dintwa. An overview of solar power (pv systems) integration into electricity grids. *Materials Science for Energy Technologies*, 2(3):629–633, 2019.

[17] The Sejm of the Republic of Poland. Rozporządzenie ministra gospodarki z dnia 4 maja 2007 r. w sprawie szczegółowych warunków funkcjonowania systemu elektroenergetycznego. dz.u. 2007 nr 93. poz. 623. https://isap.sejm.gov.pl/isap.nsf/DocDetails.xsp?id=WDU20070930623.

[18] The Sejm of the Republic of Poland. Ustawa z dnia 20 lutego 2015 r. o odnawialnych źródłach energii. dz.u. 2015 poz. 478. https://isap.sejm.gov.pl/isap.nsf/DocDetails.xsp?id=WDU20150000478.

[19] The Sejm of the Republic of Poland. Ustawa z dnia 27 stycznia 2022 r. o zmianie ustawy o odnawialnych źródłach energii oraz ustawy o zmianie ustawy o odnawialnych źródłach energii oraz niektórych innych ustaw. dz.u. 2022 poz. 467. https://isap.sejm.gov.pl/isap.nsf/DocDetails.xsp?id=WDU20220000467.

[20] Jonathan J Oliver, Rohan A Baxter, and Chris S Wallace. Unsupervised learning using mml. In *ICML*, pages 364–372. Citeseer, 1996.

[21] FY Osisanwo, JET Akinsola, O Awodele, JO Hinmikaiye, O Olakanmi, and J Akinjobi. Supervised machine learning algorithms: classification and comparison. *International Journal of Computer Trends and Technology (IJCTT)*, 48(3):128–138, 2017.

[22] N.L. Panwar, S.C. Kaushik, and Surendra Kothari. Role of renewable energy sources in environmental protection: A review. *Renewable and Sustainable Energy Reviews*, 15(3):1513–1524, 2011.

[23] P. Parczyk. Remote solar system telemetry. Bachelor's thesis, Wrocław University of Science and Technology faculty of Electronics, 2020.

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

[25] Larry Schumaker. *Spline functions: basic theory*. Cambridge University Press, 2007.

[26] SciPy. Scipy interpolation. https://docs.scipy.org/doc/scipy/tutorial/interpolate.html, 2022.

[27] Soleopv.pl. How does the pv power plan work? https://soleopv.pl/blog/jak-dziala-instalacja-fotowoltaiczna/, 2019.

[28] Katarzyna Stapor, Paweł Ksieniewicz, Salvador García, and Michał Woźniak. How to design the fair experimental classifier evaluation. *Applied Soft Computing*, 104:107219, 2021.

[29] Yuzuru Ueda, Kosuke Kurokawa, Takayuki Tanabe, Kiyoyuki Kitamura, Katsumi Akanuma, Masaharu Yokota, and Hiroyuki Sugihara. Study on the over voltage problem and battery operation for grid-connected residential pv systems. In *22nd European Photovoltaic Solar Energy Conference*, pages 3094–3097, 2007.

[30] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[31] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[32] Grady Barrett Wright. *Radial basis function interpolation: numerical and analytical developments*. University of Colorado at Boulder, 2003.

[33] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. 2005.

[34] P. Zyblewski. *Classifier selection for imbalanced data stream classification*. PhD thesis, Wrocław University of Science and Technology, 2021.