UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Mehdi Hatamian

Predicting Location-Based Green Energy Availability in Smart Buildings

Master's Thesis (30 ECTS)

Supervisor: Chinmaya Kumar Dehury, Ph.D.

Tartu 2022

Predicting Location-based Green Energy Availability in Smart Buildings

Abstract:

Today, renewal energies have been gaining more attention across the globe due to their clean energy production. Solar energy is the most abundant renewable resource of energy on earth. Solar power represents a clean and green energy source in the energy transition era. Accordingly, the photovoltaic (PV) solar panel system is the most common way in which solar energy is captured. Companies that generate energy need to predict the amount of energy sold in the electricity pool day-ahead or intra-day to maintain power production and demand in balance. Additionally, the Return on Investment (ROI) is what everyone wants to know for investing in PV solar panels. Therefore, one of the concerns about solar panels is their efficiency due to the low reliability of certain renewable sources, including the variation of the weather conditions. Solar panels are highly dependent on how much sunlight they receive, and it's difficult to predict the amount of power generated by solar panels. Thus, reducing uncertainty is a solution by an energy forecasting tool that can predict the output of solar panels throughout the year. In this research, a detailed procedure is proposed to forecast the output power of PV solar panels by different machine learning models. The goal is to achieve a best-fitting model which is more accurate by inspecting the data precisely. Several predictive models will be compared to identify the best and most suitable ones for the described case.

Keywords:
Solar Panel, Smart Building, Green Energy Prediction, Solar Panel, and Weather, Machine Learning, Output Power Prediction

CERCS: P170 Computer science, numerical analysis, systems, control

Asukohapõhise rohelise energia kättesaadavuse ennustamine nutikates hoonetes

Lühikokkuvõte:

Tänapäeval on uuenemisenergiad pälvinud kogu maailmas rohkem tähelepanu tänu nende puhtale energiatootmisele. Päikeseenergia on kõige rikkalikum taastuv energiaallikas maa peal. Päikeseenergia on energia ülemineku ajastul puhas ja roheline energiaallikas. Sellest lähtuvalt on fotogalvaaniline päikesepaneelide süsteem kõige levinum viis päikeseenergia kogumiseks. Energiat tootvad ettevõtted peavad energiatootmise ja -nõudluse tasakaalus hoidmiseks prognoosima elektribasseinis müüdava energia kogust päev ette või päeva jooksul. Lisaks on investeeringutasuvus see, mida kõik tahavad teada fotogalvaanilistesse päikesepaneelidesse investeerimise kohta. Seetõttu on päikesepaneelide üheks murekohaks nende tõhusus, mis on tingitud teatud taastuvate energiaallikate vähesest töökindlusest, sealhulgas ilmastikutingimuste muutumisest. Päikesepaneelid sõltuvad suuresti sellest, kui palju päikesevalgust nad

saavad, ja päikesepaneelide toodetud võimsust on raske ennustada. Seega on ebakindluse vähendamine lahendus energia prognoosimise tööriista abil, mis suudab ennustada päikesepaneelide toodangut aastaringselt. Selles uuringus pakutakse välja üksikasjalik protseduur fotogalvaaniliste päikesepaneelide väljundvõimsuse prognoosimiseks erinevate masinõppemudelite abil. Eesmärk on saavutada kõige paremini sobiv mudel, mis oleks täpsem andmete kontrollimisel. Võrreldakse mitmeid ennustavaid mudeleid, et selgitada välja kirjeldatud juhtumi jaoks parimad ja sobivaimad.

Võtmesõnad:
Päikesepaneel, nutikas hoone, rohelise energia prognoosimine, päikesepaneel ja ilm, masinõpe, väljundvõimsuse prognoosimine

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

1 Introduction

Technology advancements and government policies have strongly encouraged and supported the generation of Renewable Energy (RE) [4]. Thus, producing energy by solar irradiation is considered to be the most promoting and safe energy supplied from the PV systems [5]. A unit of irradiance is the power density of sunlight or the total output of a radiant source falling on a unit area. Solar irradiance determines the amount of power produced on a particular day. However, irradiance depends upon various factors, including location, weather, time, etc. Increasing solar system installation offers a safe method of reducing carbon dioxide emissions in the "energy transition era." PV solar panel system is the most common way in which solar energy is captured. Companies that generate energy need to predict the amount of energy sold in the electricity pool day-ahead or intra-day to maintain power production and demand in balance. Additionally, the ROI is what everyone wants to for investing in PV solar panels. Therefore, one of the major challenges regarding solar panels is their efficiency due to the low reliability of certain renewable sources, including a variation in the weather conditions. Additionally, solar radiation is crucial in the design of a photovoltaic system. The power produced by solar cells has a direct relationship to the current local weather condition and varies throughout the day as the amount of solar irradiance changes [3]. Thus, the power generated by the solar panel system is highly dependent on weather conditions that might result in unstable output energy.

The generated power is not easily predictable in advance. Thus, their power generation is intermittent and uncontrollable [1]. Furthermore, the prediction of the amount of electricity that solar panels will generate is essential for the calculation of the size of the system, ROI, and system load measurements [2]. Numerous methods have been developed in the literature to predict the output power of PV systems. The prediction methods are classified into model-based and data-driven [6]. Analytical equations form the basis of model-based methods that focus on PV power production. The output power is predicted by the equations leveraging weather conditions [7].

Furthermore, machine learning techniques have gained popularity in recent years to predict the output power of the solar system. The main objective of this research is to perform different machine learning techniques to compare their performance by predicting the output power of solar panels. Machine learning algorithms are parameterized. Therefore, the behavior of the machine learning models can be tuned for a given problem. This research predicts output power using the state-of-art machine learning models, such as Extreme Gradient Boosting (XGboost), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Multi-Layer Perceptron (MLP), Random Forest Regressor (RF). The selection of these models was based on their tendency to perform well in previous research of energy forecasting. The goal is to give a general overview of analyzing the methods' performance rather than studying a specific model in detail. Predictions are evaluated by using popular error rate methods, such as Mean Absolute Percentage

Error (MAPE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Coefficient of Determination (R2).

The rest of this research is organized as follows. In section 2 an overview of general research and information about the state of the art literature is presented. Data processing, feature selection, outlier handling, feature scaling, and target transformation are discussed in section 3 with the methodology that is utilized. All used machine learning models in this research are discussed in section 4, including XGBoost, RF, KNN, MLP, and SVR. In order to evaluate the prediction of the models, different performance metrics are explained in section 5. Moreover, in section 6, Hyperparameter optimization is discussed, and the methods such as manual searching, grid search, and random search are proposed. Additionally, in section 7, implementation and the results are presented followed by conclusion in section 8.

2 Literature Survey

An overview of general research and information about the state of the art literature in the research field about energy forecasting is presented in this section. Specifically, machine learning-related articles are demonstrated.

2.1 General Research

A literature study was conducted by Inman et al. [8] and Antonanzas et al. [9]. These articles provide a broad understanding of Theories and forecasting techniques related to solar PV panels. Tserenpurev et al. [10] summarizes the various forecasting techniques used in the field of solar power based on weather and air pollution features separately. This article compares different methods such as SVR, MLP, KNN, Random Forest, and Gradient Boosting. In this research paper, different sources are proposed for the predictions based on solar panel features, weather features, and air pollution features. The author found better accuracy with the Random Forest over the other models. J.Barrera et al. [11] conducted a study where Artificial Neural Network (ANN) is used to analyze how different factors affect the prediction of energy production. Compared to other literature proposals, this article has gained more accuracy by evaluating models based on lower MSE. A.Saberian et al. [12] leveraging five years of data to predict output power using ANNs in which two neural network structures are used; General Regression Neural Network (GRNN) and Feed-forward Back Propagation(FFBP). The author concludes that while both have shown a good modeling performance, FEBP performs better.

Nageem et al. [13] compare the SVR model with an analytical model. The generated output power can be calculated by location, solar panel orientation, and solar irradiance in the analytical method. Hourly measured parameters are given to the models as inputs. Additionally, Mean Absolute Error(MAE) is obtained for each month separately. The author found that the accuracy is slightly less than SVR when using analytical methods regarding the comparative study.

De Leone et al. [14] proposed SVR to forecast the output power of PV systems. This method uses past meteorological data and output power to forecast future output. The author concludes that the quality of the predicted output power depends heavily on the accuracy of weather data.

Persson et al. used Gradient Boosted Regression Trees (GBRT) to predict output power. Historical meteorological features, as well as output power data, were used. Regarding RMSE, GBRT performs better than the adaptive recursive linear (AR), time-series model, persistence model, and climatology model.

Shi et al. [15] conducted a study to use an algorithm for weather classification and Support Vector Machine (SVM) to predict output power. The study is based on the classification of the weather data for correlation analysis on the local weather forecast

and the output power. SVM with Radial Basis Function (RBF) kernel was fitted to the different weather classes to find a way to train data on specific climatic conditions.

Theocharides et al. [16] compared different machine learning models, namely, ANNs, SVR, and Regression Trees (RTs). The authors examine different hyper-parameters and set of features to predict solar panels' output power. Accordingly, the result shows that ANNs outperform other models when predicting output power.

khademi et al. [18] conducted a study on MLP with an Artificial Bee Colony (MLP-ABC) algorithm. In this study, the weather was split into sunny and cloudy days. Therefore, the author concluded that splitting data into different weather conditions enhances the accuracy of the output predictions.

2.2 Conclusions and Insights

There are mainly two approaches for predicting solar panels' output power: statistical approach and physical approach. Statistical approaches are made by leveraging historical time series, and the physical approach is made by using weather data [3]. The most widely used physical method for predicting output power is based on an appropriate theoretical model [20] in which weather conditions are in priority. Almost all studies have shown the importance of meteorological weather data, and models are sensitive to environmental situations. One solution proposed by [17] is to use different sources for Numerical Weather Prediction (NWP). In [18], the author also suggests splitting the weather dataset into different weather conditions. Therefore splitting data into different weather conditions results in more accuracy. The main method that almost every article emphasizes is the importance of non-linear models because of their ability to generalize better.

Finally, the most relevant research papers are selected regarding machine learning models that they utilized, used features, performance metrics, and time intervals for training data. The most used features are shown in Table 1 where the most widely used features are temperature, wind speed, global horizontal irradiance, humidity, respectively. Snowfall, precipitation, daylight, and sunlight are not used in any research paper. Moreover, the used machine learning models are presented in Table 2. The most widely used model is ANN, followed by SVR and regression trees, respectively. Additionally, the results from the selected research papers are shown in Table 3. In the first research papers by [10] the best accuracy has been achieved by RF. In study conducted by [6] [11] [16] the best accuracy has achieved by ANN. For the rest of the research papers, SVR is the only model that has been discussed that shows good accuracy, especially in the research study by De Leone et al. [14].

Table 1. Most Popular Features Used by Different Research Papers

		Research Papers					
		Tserenpurev et al. [10]	Al Dahidi et al. [6]	Barrera et al. [11]	Nageem et al. [13]	Theocharides et al. [16]	De Leone et al. [14]
Features	Tilt Angle and Orientation	✓	×	✓	×	×	×
	Module temperature	✓	×	×	×	×	×
	Direct normal irradiance	×	×	✓	×	×	×
	Diffuse horizontal irradiance	×	×	✓	✓	×	×
	Global horizontal Irradiation	✓	✓	×	×	✓	✓
	Reflected Irradiation	×	×	✓	×	×	×
	Humidity	✓	✓	×	✓	✓	×
	Sunshine	✓	×	×	×	×	×
	Cloud Coverage	✓	×	×	×	×	×
	Temperature	✓	✓	✓	✓	✓	✓
	Wind speed	×	✓	✓	✓	✓	×
	Snowfall	×	×	×	×	×	×
	Precipitation	×	×	×	×	×	×
	Atmospheric pressure	×	×	×	✓	×	×
	Sunlight	×	×	×	×	×	×
	daylight	×	×	×	×	×	×
	Wind direction	×	×	×	×	✓	×
	Azimuth angle	×	×	×	×	✓	×

Table 2. Most Popular ML Methods Used by Different Research Papers

		Research Papers					
		Tserenpurev et al. [10]	Al Dahidi et al. [6]	Barrera et al. [11]	Nageem et al. [13]	Theocharides et al. [16]	De Leone et al. [14]
Models	SVR	✓	×	×	✓	✓	✓
	MLP	✓	×	×	×	×	×
	kNN	✓	×	×	×	×	×
	RF	✓	×	×	×	×	×
	GB	✓	×	×	×	×	×
	Linear Regression	✓	×	×	×	×	×
	ANN	×	✓	✓	×	✓	×
	Regression Tree	×	✓	✓	×	✓	×

Table 3. The Results of the Most Popular ML Methods

		Research Papers					
		Tserenpurev et al. [10]	Al Dahidi et al. [6]	Barrera et al. [11]	Nageem et al. [13]	Theocharides et al. [16]	De Leone et al. [14]
Results	RMSE	2.38	10.011	0.2			0.99
	MSE			0.04			
	MAE	1.38	7.169	0.161			
	R2	0.87					0.95
	MAPE				0.36	0.6	0.35

3 Data and Problem Description

This section includes the research methodology of the thesis. Thus, data acquisition, data processing details, feature selection, outlier handling, feature scaling, and target transformation are under assessment. This research presents a comparative analysis between different machine learning algorithms to choose the most accurate one that performs well in the field of PV solar panels.

3.1 Data Processing

The data of output power was collected from one source between 20-9-2020 to 27-9-2021, and the information about the data is given in Table 4 by the University of Tartu [90]. This research is a real case study of a solar PV system mounted on the rooftop of the Delta Center-University of Tartu building in Tartu, Estonia Fig. 1. (latitude = 58.38548, Longitude= 26.72475).

Figure 1. Delta Center PV Map (retrieved and adapted from Google Maps. [19]

Table 4. Information about Output Power

Dataset Characteristic	Time Series	Number of Instances	35730
Attribute Characteristic	Continuous, Floats	Number of Attributes	2
File Format	xlsx	Missing Values	No
File Size	841 KB	Date Format	DD/MM/YYYY
Time Format	HH:MM:SS XM	Area/Field	Solar Panel

The real dataset consists of real weather data measured by the Tartu-Tõravere weather station located far away by around 5Km from the Delta Center building. The weather-related data is collected from the Republic of Estonia Environment Agency [89]. The information about the weather data is given in Table 5 and the description of collected features is provided in Table 6.

Table 5. Information about Weather Data

Dataset Characteristic	Multivariate, Panel Data	Number of Instances	8928
Attribute Characteristic	Continuous, Floats, Integer	Number of Attributes	11
File Format	xlsx	Missing Values	No
File Size	1.04 MB	Date Format	DD/MM/YYYY
Time Format	HH:MM:SS XM	Area/Field	Weather

The data about the output power of PV solar panels are provided by the University of Tartu in 15-minutes intervals. The data have been relatively clean after merging the separated datasets. Additionally, the set of features extended by using derived features including "Month" and "Hour" in order to gain more accuracy when training models.

3.2 Feature Selection

The overall output of a PV solar panel depends on many factors, including the performance of the panel, weather condition, operating condition, and so on [64]. When many factors are taken into consideration as input features, the accuracy may be reduced.

Table 6. Historical Weather Data Descriptions

Type	Feature	Description	Unit	Range
Dependent	Power output	Power output of solar panels	kWh	Unlimited
Independent	Cloud Coverage	Cloud Amount	Okta	0 - 9
	Average Air Pressure(sea level)	Pressure within the atmosphere of earth	Milibars	100-1050
	Amount of precipitation	Any product of the condensation of atmospheric water vapor	Millimeter	Unlimited
	Relative Humidity	Ratio of how much water vapor is in the air	Percentage	0 - 100
	Amount of sunshine	Direct sunlight duration without being covered by clouds	Minutes	0-60
	Air temperature	Temperature	Celsius	Unlimited
	Wind direction	Air Direction	Degree	0-360
	Average wind speed	Air movement	Mile per second	Unlimited
	Maximum wind speed	Air movement	Mile per second	Unlimited
	Global horizontal irradiance	The total solar radiation incident on a horizontal surface	Watts per square meter	NA

Thus, a method must be developed to determine the correlation between various factors and photovoltaic output power. As part of machine learning, feature selection is used to select a subset of relevant features for modeling purposes. The feature selection technique removes redundant or irrelevant features from the data, as well as features that are strongly correlated without much information loss [86]. Feature selection is required in order to reduce runtime, eliminate unwanted noise, and features that might mislead the predictions. However, in this study, KNN, SVR, and MLP are the main models that require a feature selection procedure to reduce the runtime and prevent overfitting.

3.2.1 Pearson Correlation

The Pearson correlation coefficient is commonly used in statistics when measuring the linear correlation between two variables X, and Y [65] [66]. Pearson correlation analysis is used to evaluate the data to find a relationship between two or more variables. The correlation value is calculated by the given Eq. 1 [82]:

$$\text{correlation}\left(f_i, f_j\right) = \left| \frac{\sum_{d\epsilon|docs|}\left(f_{i,d} - \bar{f}_i\right)\left(f_{j,d} - \overline{f_j}\right)}{\sqrt{\sum_{d\epsilon|docs|}\left(f_{i,d} - \bar{f}_i\right)^2}\sqrt{\sum_{d\in|docs|}\left(f_{j,d} - \bar{f}_j\right)^2}} \right| \tag{1}$$

where $\bar{f}_i$ and $\bar{f}_j$ are mean values of f_i and f_j vector. The values of feature i and j is represented by $f_{i,d}$ and $f_{j,d}$ for dth document.

Moreover, the linear relationship between output power and other features is provided in Fig. 2 by using the Pearson correlation coefficient. A Pearson correlation is determined by its degree of strength that varies between a strong to a weak correlation or a zero correlation. By this correlation, 1 represents a strong positive correlation, and -1 represents a strong negative correlation. Additionally, zero means no correlation between features. As can be seen in Fig. 2, regarding the correlation coefficient values for solar irradiance versus output power, it can be said there is a positive relationship between these parameters.

Furthermore, there is a positive relationship between sunshine amount versus output power as well as temperature versus output power. However, there is a negative relationship between relative humidity versus output power and cloud coverage versus output power.

3.2.2 Feature Importance

A feature importance score is calculated for a given model using the input features. Each score represents the importance of a particular feature. The higher the score, the more influence the specific feature will have on predicting the final output. The feature

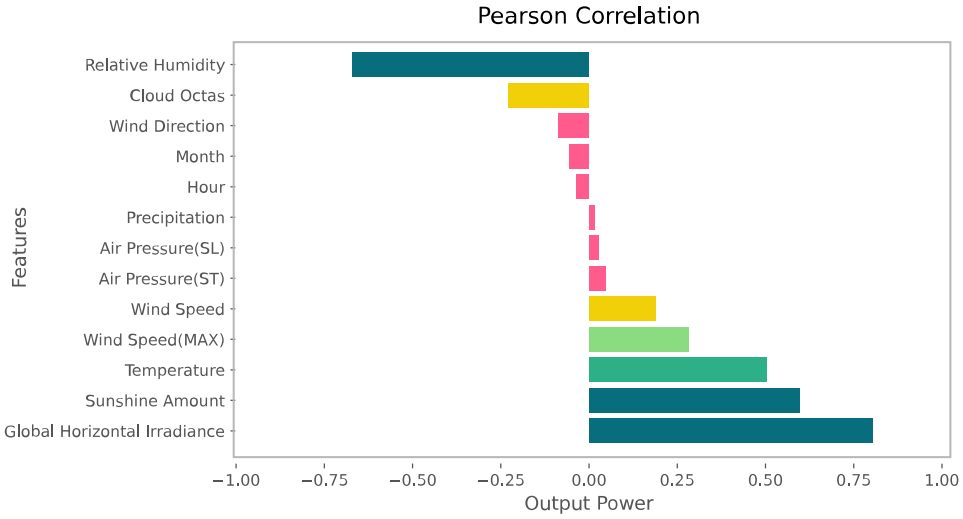

Figure 2. Pearson Correlation

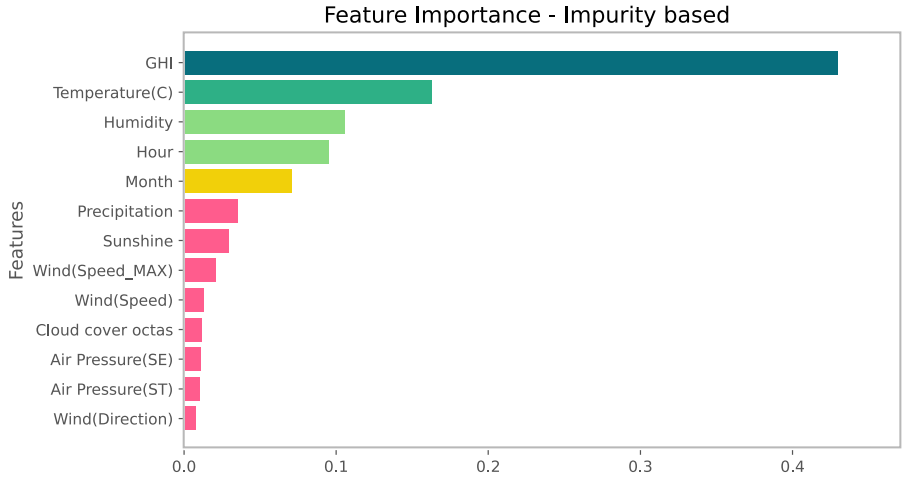

Figure 3. Feature Importance by Gini Impurity

selection procedure based on feature importance utilizes the XGBoost algorithm and Lasso regression to minimize the cost function.

- **XGBoost Feature Importance:** This approach works on the principle of Gini impurity that considers each input data parameter to determine how much each parameter minimizes cost function [17]. MSE is used as an impurity measure

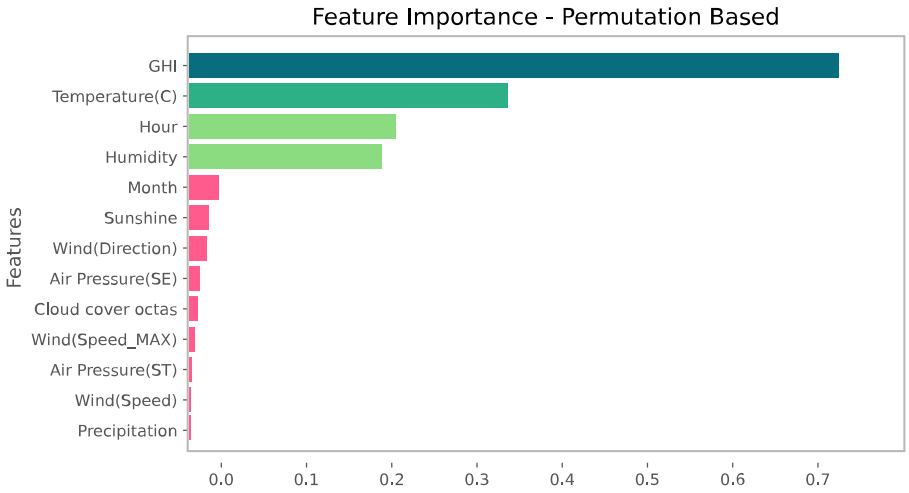

Figure 4. Permutation Based Feature Importance

in regression to split the feature. Therefore, the calculated score from feature importance can be used to reduce the dimensionality of the model during training. However, feature importance by leveraging Gini impurity is potentially a biased approach since it tends to inflate the importance of continuous features or high-cardinality categorical variables [83]. More specifically, when features are correlated, this approach can select one of the features and ignore the importance of the second one. Therefore, permutation-based feature importance is also included in Fig. 4. In this method, each feature is shuffled, and the change in the model's performance is computed. Accordingly, this method overcomes the drawbacks of calculated feature importance by Gini impurity.

- **Lasso Regression Feature Importance:** Another approach for feature selection is known as lasso regression, which utilizes a regularization technique. Lasso regression has a penalty term. Adding a penalizing term to a normal regression generates a Lasso regression to avoid overfitting. More specifically, lasso regression shrinks the values of the unimportant variables to zero, in which feature selection is already included. Therefore, data values are shrunk towards the mean. Lasso regression tends to assign zero weights to most irrelevant or redundant features, and hence is a promising technique for feature selection [87]

  Lasso regression is fitted on the scaled version of the dataset. Accordingly, those features that are different from zero will be considered in our training.

3.3 Outlier Handling

Regarding solar panels, different factors can degrade output power and different behavior, resulting in anomalous parameters. There are outliers in a dataset when values are at the extreme ends. Outliers are observations that appear to be inconsistent with the rest of the data [61]. Despite the fact that some outliers reflect true values from natural variation, others result from errors. Outlier detection is required to reduce the negative influence on the model accuracy.

According to [62], when dealing with outliers, the term "appears to be inconsistent" is the main problem, and that is what outlier detection methods attempt to address. Additionally, different approaches can detect outliers, including statistical, probabilistic, Bayesian techniques, distance-based, etc. If we know the data distribution, many statistical approaches work well.

The mean and standard deviation have been widely used by statisticians to identify outliers [61]. Furthermore, Under PV faults at relatively high contamination levels, the Boxplot rule shows the best accuracy and robustness [73]. An illustration of the boxplot is given in Fig. 5.

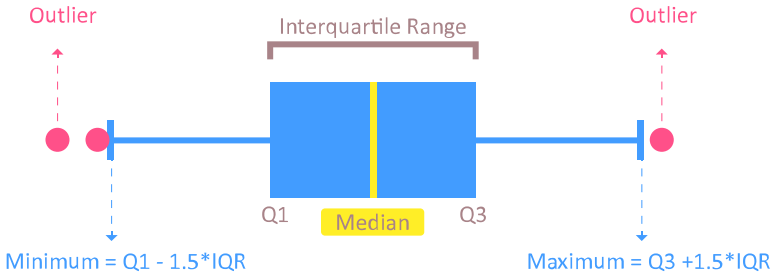

Figure 5. Boxplot Outlier Detection

After visually inspecting the dataset and regarding the skewed distribution of data, an Interquartile Range (IQR) proximity rule was used to find the spread of distribution by IQR.

According to the IQR rule, when a value falls outside boundaries given in the Eq. 2, it is considered an outlier [21].

$$\begin{cases} \text{Lowerboundary} = Q_1 - 1.5IQR \\ \text{Upperboundary} = Q_3 + 1.5IQ \\ \text{IQR} = Q_3 - Q_1 \end{cases} \tag{2}$$

First, the data is required to be sorted. Then four equal parts (5) will be derived to find the distance between two middle sets of data.

3.4 Feature Scaling

Feature scaling would be an essential task to control the variability of the dataset. Feature scaling of data is a way of normalizing the range of independent variables. Due to the fact that the features might not be in the same space (or unit), a normalization rule must be considered. There are mainly two of the most commonly used approaches to make all features on the same scale:

- **Standardization:** A scaling method that makes all values to be centered around the mean with a unit standard deviation. When outliers are important, then using standardization is a key point to handle outliers. The transformed variable will have a mean of 0 and a variance of 1 as given in the Eq. 3 [74]:

$$x_{ij} = Z\left(x_{ij}\right) = \frac{x_{ij} - \bar{x}_j}{\sigma_j} \tag{3}$$

  Where x and σ represent the sample mean and standard deviation, respectively.

- **Normalization:** The normalization process involves shifting and re-scaling values so that they end up ranging between 0 and 1. Min-Max scaling is also known as normalization. The feature X is considered with n observations, thus a common normalization for each $X_i, 1 \leq i \leq n$ is given by the Eq. 4 [17].

  Min-Max scaling involves moving the values towards the mean of the column. Therefore, if outliers are significant regarding their impact, z-score normalization is considered better.

$$X_i' = \frac{X_i - \min X}{\max X - \min X}, \quad i \in 1, \ldots, n \tag{4}$$

3.5 Target Transformation

After visually inspecting the data and calculating the skewness, we can conclude that the distribution of the output power as the dependent feature is right-skewed, as shown in Fig. 6. The illustrated distributions represent the frequency of the data that is right-skewed distribution, and the number of times a data value occurs is the frequency of the data. Right skewed is recognized as positive skew since most values are clustered in the left tail while the right tail is longer on the right side. Skewness measures the symmetry [84] to understand the distortion that deviates from the normal distribution. The Eq.5 is given:

$$\gamma(X) = \frac{n}{(n-1)(n-2)} \sum_{x \in X} \left(\frac{x - \bar{x}}{\sigma} \right)^3 \tag{5}$$

Where n represents the number of values, mean is given by $\bar{x}$ and sigma is represented by σ.

Skewness was calculated by using the skew function provided by SciPy [85]. Data is highly skewed if the skewness value is above or below +1 or -1. A moderately skewed distribution lies between -1 and -0.5 or between +0.5 and 1. If the skewness indicates a value between -0.5 and 0.5, the data distribution is nearly symmetrical. The data is symmetric when the value is 0 [88]. The skewness of the output power is given in Table 7.

Table 7. Skewness of Output Power

Skewness Before Removing Zeros	2.4
Skewness After Removing Zeros	1.02
Skewness After Applying Transformation	0.36

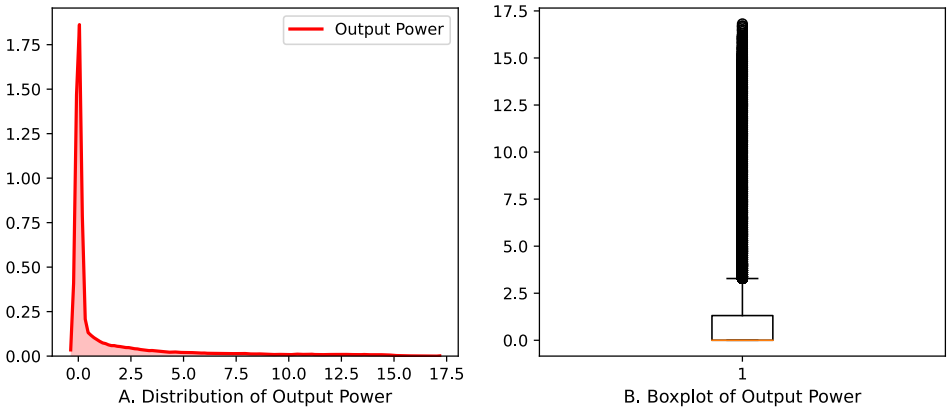

Figure 6. Right Skewed Distribution Including Zeros

The first step is to temporarily exclude all zeros to detect outliers in actual data as shown in 7. The functionality of solar panels is depended on the light. Thus, temporary removing nighttime data from output power is necessary. However, we will include them in our final predictions since nighttime data allow us to use the whole day for identifying the time regions. Therefore, square root transformation is applied to make the distribution more normalized, as is illustrated in Fig. 8.

After inspecting the presence of outliers and ensuring that the actual data is not affected by outliers, the transformation will be applied to the output power that all zeros are included. The final distribution for training data is illustrated in Fig. 9. Accordingly,

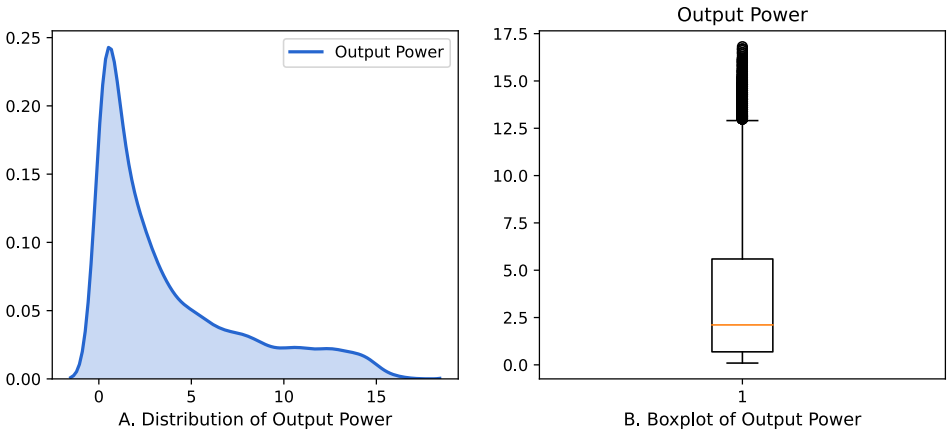

Figure 7. Right Skewed Distribution After Removing Zeros

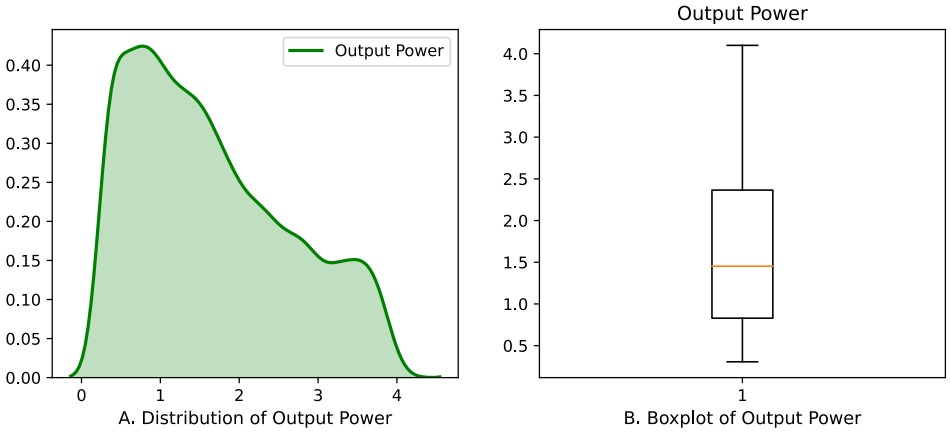

Figure 8. Distribution of Output Power After Transformation (Zeros Are Excluded)

square root transformation stabilizes the distribution variance that could help decrease the skewness, and transformation penalizes higher values more than smaller ones.

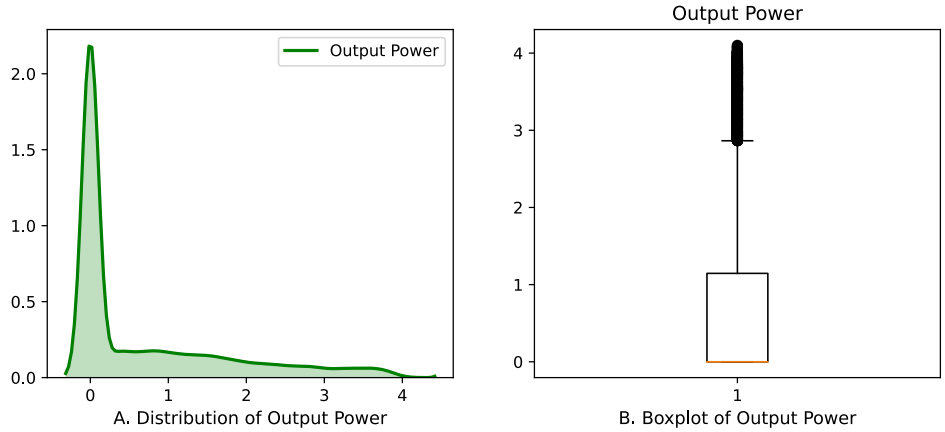

Figure 9. Distribution of Output Power After Transformation (Including Zeros)

4 Machine learning models

The output power prediction uses state-of-art machine learning models such as XGboost, KNN, MLP, RF, and SVR. Additionally, different metrics are used to evaluate their performance, such as Mape, MAE, RMSE, and R-squared. In this study, machine learning techniques are compared for the output power of photovoltaic panels by historical weather data. Among tree-based algorithms, RF and XGBoost are preferred. According to previous research, RF tends to produce high accuracy. Furthermore, XGBoost was taken into consideration since it is not popular in output power forecasting but gained popularity for its performance among tree-based algorithms. Moreover, XGBoost is utilized to compare the results of RF as a base model. ANN has always shown great results in the field of forecasting, and KNN was taken into consideration because the historical weather data is highly correlated to their neighbors, which results in correlated information in a specific feature space. Consequently, KNN was selected because the neighbors closer to each other contribute more to the average compared to those further away. Finally, SVR was discussed in order to the probability of non-linearity in meteorological data since its performance is superior in nonlinear modeling and its ability to generalize better.

4.1 Extreme Gradient boosting - XGBoost

The concept of XGBoost is derived from Gradient boosting, which implements a Gradient Boosting algorithm based on decision trees. Boosting is an ensemble learning technique. Ensemble learning involves training multiple models as the weak learners in which weak learners are combined to get more accurate predictions [28]. Ensemble methods improve the performance of multiple existing models. Moreover, the ensemble method

depends on randomization techniques and results in many solutions to the problem [26]. Accordingly, this procedure yields boosting. Boosting refers to producing a very accurate prediction rule. Boosting algorithms create weak learners sequentially that have a weak correlation with the actual data. The goal is to find a predictive function $F^*(X)$ [17]:

$$F^*(X) = \arg\min_{F(X)} E_{Y,X} L(Y, F(X)), \tag{6}$$

L is the loss function. Additionally, $F^*(X)$ is approximated by:

$$F(X) = \sum_{m=0}^{M} \beta_m h(X; \bar{a}_m) \tag{7}$$

where $h(X; \bar{a}_m)$ is a weak leaner with coefficients $\bar{a}_m$ and the total number of weak learners is M.

Furthermore, a base learning algorithm as a weak learner is required to make an accurate prediction.

Boosting learns from mistakes made in previous predictions and corrects them in the next predictor. Leslie Valian in [23] states that the idea of boosting algorithms is to exploit the weak learning method repeatedly in order to get a succession of hypotheses. Thus, each one refocuses on the examples that the previous ones found difficult. Accordingly, a weak learner transforms into a stronger one in an iterative way.

Exploiting base learning algorithms such as tree-based or linear-based models is the key point when weak rules are generated at each iteration. In boosting, a weak learner is referred to as a base model.

4.1.1 Adaptive Boosting

Adaptive boosting or AdaBoost is the first boosting algorithm. AdaBoost works by putting more weight on samples that are difficult to classify until the model identify a correct classify method as given in Eq. 8 [24].

AdaBoost uses multiple trees combined as weak learners to output the final result. Each weak learner is fitted on bootstrapped data. Accordingly, the weight of each weak learner is adjusted regarding residual errors [29].

$$F_i(x) = F_{i-1}(x) + f_i(x) \tag{8}$$

where $F(i)$ represents current model, $F(i-1)$ is the previous model and $f(i)$ is the weak learner.

4.1.2 Gradient Boosting

Gradient boosting is another boosting algorithm. Gradient boosting is used for both classification and regression predictive models as a class of ensemble machine learning algorithms. Gradient boost identifies large residuals in the previous prediction to minimize loss function in the next prediction. Fig. 10 represents the Gradient boosting algorithm.

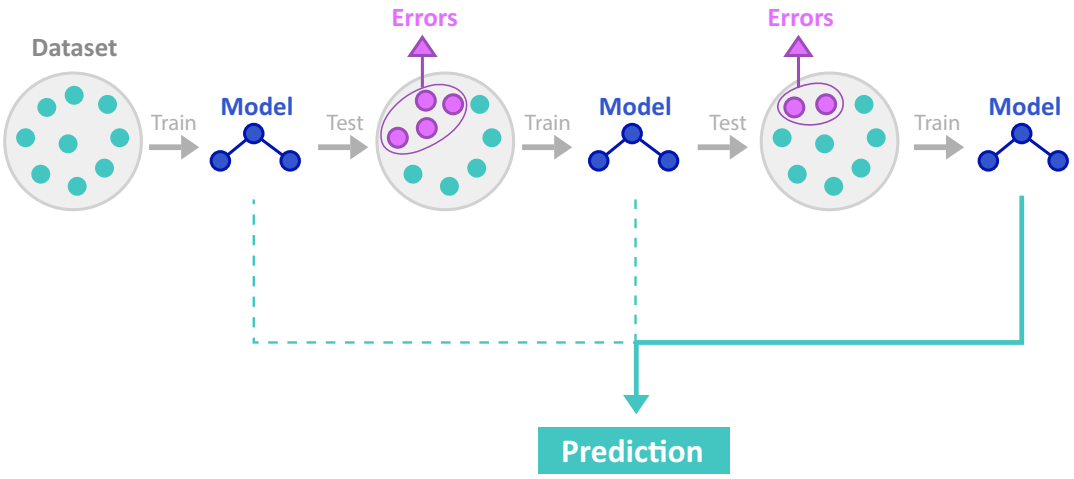

Figure 10. Gradient Boosting

An additive manner is required by using Gradient boosting in which decision trees are added one at a time. Moreover, gradient descent is used to minimize loss function while existing trees remain without changes. Decision trees with a single split are the weak learners in Gradient boosting. Choosing the best split is determined by measuring the purity score in which the trees are constructed in a greedy manner. Additionally, weak learners are combined to reduce loss function by using gradient descent in a sequential manner to return a stronger model.

Gradient descent used in Gradient boosting minimizes a set of parameters such as the coefficients in regression equations or weights in a neural network [27]. In each iteration, the perdition for the weak learners is compared to the correct output that is expected, as is illustrated in Fig. 10.

4.1.3 XGBoost

XGBoost was initially introduced by Tianqi Chen in their 2016 paper titled "XGBoost: A Scalable Tree Boosting System." XGBoost is a scalable ensemble method based on

Gradient boosting that is improved, fast, and efficient machine learning problem solver [25].

This model utilizes advanced regularization methods such as L1 and L2, which represent lasso regression and ridge regression, respectively. Advanced regularization prevents overfitting and improves model generalization. Additionally, to optimize the objective quickly, XGBoost computes a second partial derivation of the convex loss function and offers a sparsity-aware algorithm for sparse input and a weighted quantile sketch algorithm to approximate tree learning [31]. Moreover, XGBoost Handles missing values and training can be parallelized [22].

Similarly to Gradient boosting, an additive method is required, and XGBoost minimizes loss function by building an additive expansion of objective function as given in Eq. 9 [25].

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y_i}^{(t-1)} + f_t\left(\mathbf{x}_i\right)\right) + \Omega\left(f_t\right) \tag{9}$$

Suppose the difference between the prediction $\hat{y_i}^{(t)}$ and the target y_i is to be measure by the l that is a differentiable convex loss function [22]. The complexity is penalized by the term Ω and the prediction of the i-th instance at the t-th iteration considered to be $\hat{y_i}^{(t)}$. Let f_t be the independent tree structure; thus, to improve our model adding f_t greedily is required. In fact, after calculating the loss, we must add a tree to the model to minimize the loss.

By using XGBoost, over-fitting is prevented with a regularized model [22]. Thus, some randomization techniques are used for reducing over-fitting and increasing training, including random sub-samples and column sub-samples [25].

A random sub-sample of the training data is employed at each iteration, and instead of the full training dataset, the randomly selected sub-sample is used to fit the base learner [30].

The reason that XGBoost is chosen among models is to leverage the ability of this new model for predicting the output power of solar panels. In the field of solar panels, XGBoost is not widely used, but its superior performance is the reason to utilize this machine learning model.

4.2 Random Forest (RF)

Random Forest is one of the most popular supervised machine learning algorithms for both classifications and regressions. RF is an ensemble method of decision trees. Decision trees are the main building block of RF, or in other words, a random forest is an ensemble of decision trees. In a regression ensemble model, each tree is trained independently in parallel to predict a real value. The decision tree forms a classification or regression model by asking questions starting from the root node. Thus, decision rules

based on the dataset structure are generated until leaves are reached. However, training data on decision trees are complex regarding the number of nodes created in some cases, and they can be computationally expensive for certain domains [35].

In general, RF has higher accuracy than decision trees and can handle large datasets [38].

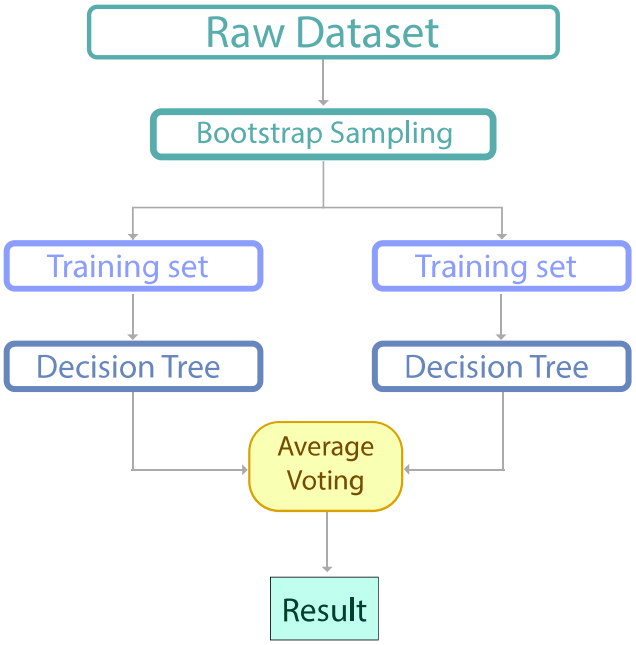

Figure 11. Random Forest Flowchart

Furthermore, RF adds randomness to the model. Therefore, unlike the decision tree wherein all the variables in the dataset are employed, in RF, the splitting variable is chosen from a random sample set of variables [36]. In other words, at each node, the selection of a random subset of features is limited. A simple flow chart of RF is shown in Fig. 11.

The RF methodology has been successfully involved in various practical problems, including air quality prediction, chemoinformatics, ecology, 3D object recognition, bioinformatics and etc. [37]. Moreover, RF is widely accepted due to its ability to handle nonlinear classification tasks efficiently [32].

RF can work well on large datasets with many features. This machine-learning algorithm involves bagging sampling approach [26] and the random selection of features to construct a collection of decision trees [34].

Biau G. [37] states that the nonparametric regression estimation is the general framework for RF, in which an input random vector $\mathbf{X} \in \mathcal{X} \subset \mathbb{R}^p$ is observed, and the goal is

to estimate the square integrable random response $Y \in \mathbb{R}$ by predicting the regression function $m(\mathbf{x}) = E[Y|\mathbf{X} = \mathbf{x}$. Additionally, with the assumption that we are given a training sample $\mathcal{D}_n = ((\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n))$ of independent random features as the independent sample pair $(\mathbf{X}, Y)$. The goal is to construct an estimate $m_n : \mathcal{X} \to \mathbb{R}$ of the function m by using the dataset $\mathcal{D}_n$.

Several major parameters for the random forest method are the number of trees and tree depth which are not sensitive and can be easily applied to the prediction of PV solar output power [38].

In many research papers, RF tends to have very good accuracy. Regarding its ability to perform well on forecasting severe weather conditions [91], we decided to use this machine learning model to compare with others.

4.3 K-Nearest Neighbour (KNN)

According to [17], KNN is a supervised and distances based classifier that estimates the conditional distribution of Y given X. Therefore, assign Y to the class with the highest estimated probability. Suppose a test observation of x_0 and the main approach to measuring the k closest points to the training data is done by Euclidean distance, represented by N_0. The conditional empirical distribution is estimated by KNN as the fraction of K nearest points that are classified as j:

$$\Pr\left(Y = j \mid X = x_0\right) = \frac{1}{K} \sum_{i \in N_0} I\left(y_i = j\right) \tag{10}$$

There are four different types of distance measurements in the KNN, including Chebyshev distance, Euclidean distance, Manhattan distance, and Minkowski distance [40].

This classifier was first studied in 1951 by Fix and Hodges in the US Air Force School of Aviation Medicine [39]. KNN predicts the output by averaging the observations in the same surrounding data points as shown in Fig. 12.

The main objective of KNN is to choose an appropriate number of neighbors (K-neighbors) to assign the instance to the class that is more similar.

The KNN performance is determined by the choice of the distance that is used. For the prediction of the power output, the Manhattan distance measurement is employed since the Manhattan distance measurement performs well on our dataset. Manhattan Distance is measured by the sum of absolute differences between points across all the dimensions. Moreover, according to the [42] the Manhattan distance metric provides higher relative contrast than the Euclidean distance metric. Furthermore, the Manhattan distance metric is the most reliable choice for providing the best contrast between the different points among the distance metrics with integral norms. The Manhattan distance measurement is given by the Eq. 11 [40]

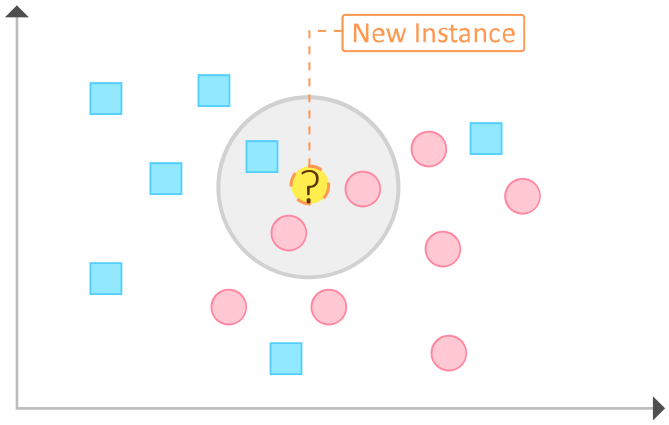

Figure 12. K-Nearest Neighbour (KNN)

$$D_{\text{Manhattan}}(x, y) = \|\mathbf{x} - \mathbf{y}\|_1 = \sum_{i=1}^{n} |x_i - y_i| \tag{11}$$

Two vectors in the feature space are represented by x and y. Additionally, x_i and y_i are their coordinates respectively. In fact, the conditional distribution of y given x is estimated.

K is usually small with a positive integer value and has a large effect on the KNN classifier. The decision boundary will overfit the training data if $K = 1$ corresponds to a classifier with low bias but high variance. In other words, our prediction becomes less stable when K is near to one, and in contrast, our predictions become more stable as we increase the value of K regarding majority voting/averaging. Using cross-validation to determine an appropriate value of K is proposed by T. Hastie J. Gareth [41]. For regression problems, KNN identifies the K-nearest neighbors by a distance metric and then assign the average value of neighbors, and the prediction is given by Eq. 12 [17]:

$$\hat{Y} = \frac{1}{K} \sum_{i \in N_0} y_i \tag{12}$$

4.4 MultiLayer Perceptron (MLP)

ANN is a computing paradigm that emulates the brain and nerve systems, which are primarily composed of neurons, and multilayer Perceptrons (MLP) is the typical neural network [50]. In fact, MLP is a class of feedforward artificial neural networks.

This machine learning model is used widely for the solution of different problems, including pattern recognition and interpolation. It was originally developed in the 1960s that is a development of neural network model [43].

In the research related to the output power of solar panels, MLP is widely used with very good accuracy.

4.4.1 Components of Neural Network

Nodes in an MLP are arranged in at least three layers: an input layer, a hidden layer, and output layer as shown in Fig. 13 in which represents a basic structure of an MLP. The hidden layer refines the input by eliminating redundant information then sending refined information to the next layer.

The elements of an ANN are artificial neurons, or processing units [1] also known as the perceptron. Neurons are located in layers and connected to each other by an adjustable connection weight that is called synaptic junction [47].

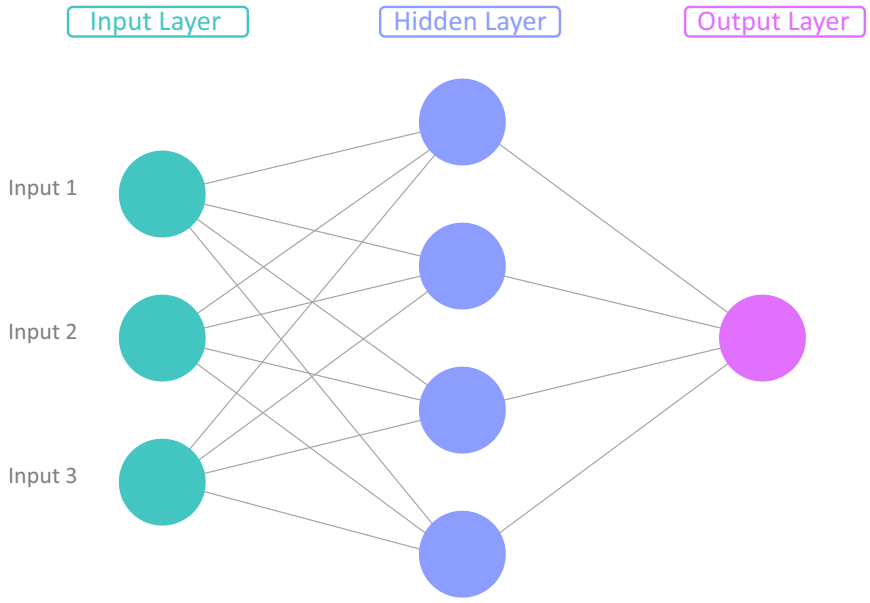

Figure 13. MLP Basic Model

MLP is made up of several layers in a single network. Every layer has an input vector, a bias vector, a weight matrix, and an output vector [46]. Perceptron or neuron takes several values as input $x_1, x_2, x_3, ..., x_n$ to produce a single output value. Fig. 15 represents a visualization of a neuron.

regardless of x-vector and y value, neurons have two other important components:

- **Weight:** The weight vector w expresses the importance of the respective inputs for the outputs; thus, weight determines how much influence the input will have on the output. A weight near zero indicates that changing this input will not have a significant effect on the output [49].

- **Bias:** Bias is a value given to the neuron as input to check whether summation is lower or higher than the bias value. To delay the triggering of the activation function, bias is used [48]. Bias increases the flexibility of a model. Although weight determines how fast the activation function will trigger and increases the steepness of the activation function, bias is the factor to prevent fast triggering of the activation function.

- **Activation Function:** In MLP, each subsequent layer multiplies the inputs with a threshold and then passes them through an activation function. Furthermore, the activation function can either be linear or nonlinear. Depending on the activation function, MLP provides clear advantages in either classification or regression. Activation functions perform complex computations are performed in hidden layers by activation functions to transfer the result to the output layer. In fact, activation functions help the network to learn complex patterns in data. Some popular activation functions are given in Fig. 14.

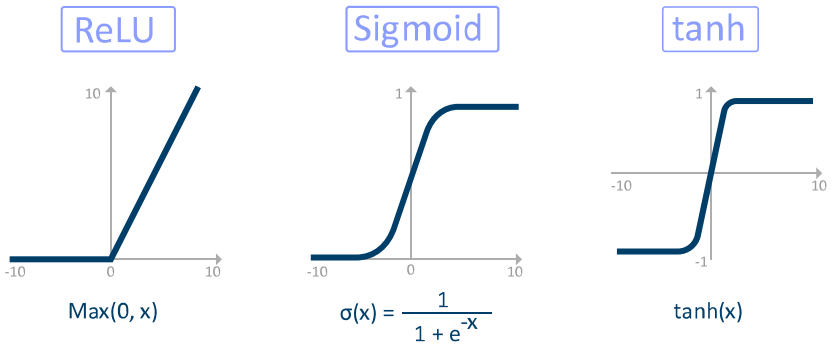

Figure 14. Activation Functions

The weighted sum based on the weight vector w is calculated to express the importance of the respective inputs for the outputs.

In [46], the author assumes the training dataset contains N points where X_p represents the p-th points from N-dimensional input data. Moreover, the output vector denotes by Y_p, and the weight W_h assigns to one of the hidden layers. Additionally, the output (L_1) from the first hidden node can be expressed as:

$$L_1(j) = \sum_{k=1}^{N+1} W_{\mathrm{h}}(j,k) X_{\mathrm{p}}(k) \tag{13}$$

Moreover, overall output is expressed as follow where $O_{\mathrm{p}}(j)$ represents the activation function:

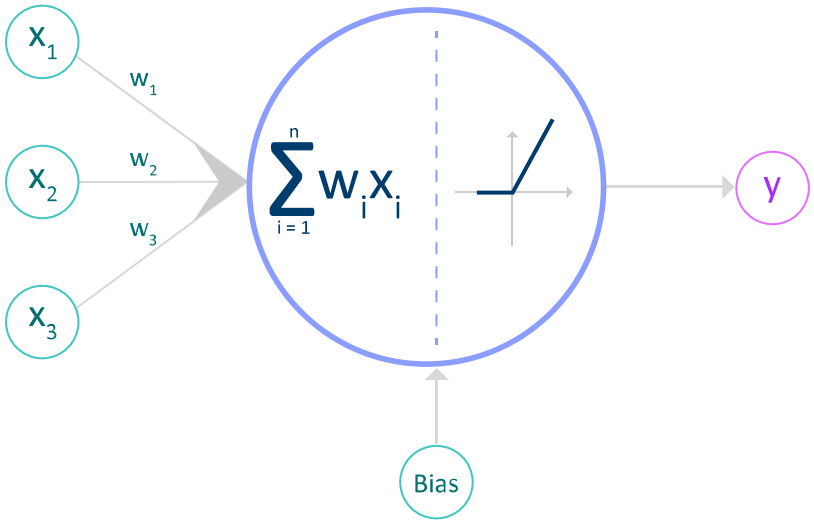

Figure 15. Visualization of a Neuron

$$Y_{\mathrm{p}}(i) = \sum_{k=1}^{N+1} W_1(i,k) X_{\mathrm{p}}(k) + \sum_{j=1}^{N_{\mathrm{h}}} W_2(i,j) O_{\mathrm{p}}(j) \tag{14}$$

In the above equation, W_1 and W_2 represent the weight from input to output and the weight from the hidden layer to the output layer, respectively.

4.4.2 Fitting a Neural Network

- **Backpropagation:** MLP works on the principle of backpropagation algorithm [44]. The backpropagation algorithm uses supervised learning in which the target and the loss are estimated by processing data in a forward direction and then in a backward direction. This procedure is done by adjusting weights from an initial guess to minimize the loss. This process continues until optimal values are reached [45]. When the data is passed through the neural network, the predicted output will be compared by the real output. Therefore backpropagation algorithm is used to minimize the loss.

- **Learning Rate:** The learning rate determines the speed at which the neural network will learn. More specifically, the size of the adjustment of the weight is determined by the learning rate each time the weights are changed during training [52]. Small values of learning rate result in small changes in weights, and large values of learning rate result in large changes in weight. A too-large value for

learning rate probably gives poor result, and a small value probably makes the training computationally time-consuming [17].

- **Optimization Algorithm:** For regression problems, the objective is to minimize the sum-of-squared errors of the model by finding the proper weight value. Stochastic Gradient Descent (SGD) is an iterative optimization algorithm for minimizing/maximizing an objective function, which tries to discover the minima or maxima by iteration. Unlike the gradient descent that requires calculation over the entire training dataset, SGD uses random sampling to pick up one example of the training set at each iteration. Moreover, SGD is most often preferred because it results in better, faster solutions [51].

4.4.3 Issues for training a neural network

- **Initial Weight:** When we train a neural network, the goal is to minimize the cost function by using a gradient descent algorithm that trains the model. Gradient descent works on the principle of finding a set of weights to best map inputs to outputs. Therefore the initial guess of the weight is important. According to [17] the network converges to an approximately linear model if weights are set to close to zero. On the contrary, the model often provides an inferior solution if choosing too large values for weight.

- **Learning Rate:** The final solution is influenced by the learning rate. Poor results can be generated by instantiating a large value of the learning rate. Furthermore, a very small learning rate value will increase computational time. To address this problem, picking a relatively large learning rate then decreasing it until improving accuracy is proposed [17].

4.5 Support Vector Machine(SVM)

Support Vector Machines (SVMs) are widely used in machine learning to solve classification and regression problems.

SVM minimizes the empirical classification error and maximizes the geometric margin simultaneously. Therefore, a maximal separating hyperplane is constructed by mapping input vectors to a higher-dimensional space. SVM utilizes the optimal hyperplane to predict the output. A hyperplane is a decision boundary that differentiates classes in an n-dimensional space that distinctly classifies the data points as shown in Fig. 16.

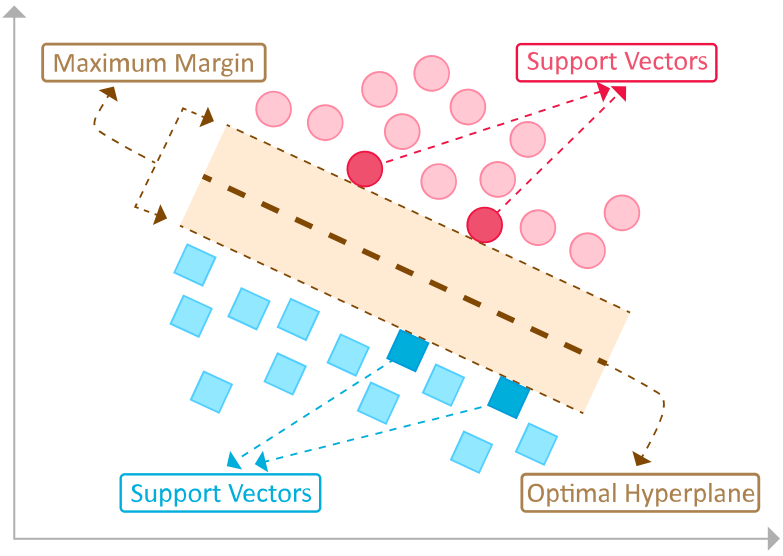

Figure 16. Support Vector Machine

4.5.1 SVM structure

- **Optimal Hyperplane:** There is usually an infinity of separating hyperplanes when a training set is linearly separable. According to [53] a separating hyperplane must be chosen that maximizes the margin between different classes or, more specifically, one that leaves as much space as possible between the hyperplane and the nearest sample. The optimal hyperplane is illustrated in Fig. 16.

- **Kernels:** The kernel is a function used in SVM to solve a nonlinear problem in order to create an optimal decision boundary by transforming the data points into the required form. Thus, the kernel transforms a low-dimensional input space into a higher-dimensional feature space.

  Choosing the right kernel function is also a research issue. However, the following kernel functions are popular for general applications [54] including the linear kernel, Gaussian RBF, sigmoid function, and polynomial function. Fig. 17 represents popular kernel functions in which γ, r and d are kernel parameters.

4.5.2 Type of Kernel Functions and kernel parameters

The linear kernel is the most basic type of kernel function and is usually in one-dimensional space. A polynomial kernel is a more generalized representation of a

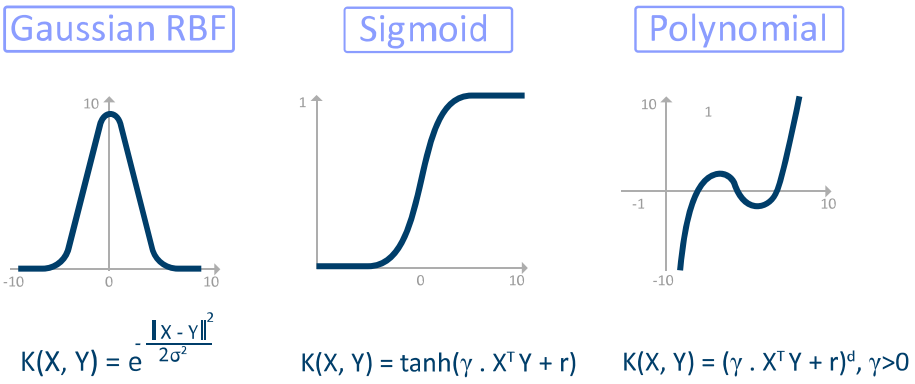

Figure 17. Kernel functions

linear kernel. According to [54] RBF is the most preferred one since it has fewer hyperparameters than the polynomial kernel, less numerical difficulties, and is used for nonlinear data. Moreover, RBF is used when the dataset is linearly inseparable. Finally, the sigmoid is the most preferred one that is usually used in a neural network. In all functions provided in 17, X and Y are referred to as the training vectors, X^T is the transposed input vector. r is a shifting parameter, and the threshold of mapping is controlled by r. When using RBF, the Bandwidth of the kernel function is controlled by σ. When the Sigma value is very small, then the decision boundaries are highly nonlinear.

Figure 18. Gamma Changes in RBF

Support vector machine with RBF kernel requires tuning these two parameters:

- **Gamma:** γ is a scaling parameter of the input data, and the decision region or spread of the kernel is determined by gamma. Fig. 18 represents the decision boundaries for different values of gamma. The model cannot capture data's "nature" or complexity when gamma is very small. Gamma determines the curvature of a decision boundary. The higher the gamma, the more curvature of the decision boundary. In fact, the larger the gamma, the narrower the kernel that results in, the more local influence of each data point.

- **Regularization parameter:** C is a coefficient error that influence of each individual point is limited by a regularization parameter. C parameter allows the model to tolerate misclassification of data points. This also refers to the regularization term. The regularization parameter prevents overfitting and controls the error. When the value of C is low, then a decision boundary with a large margin is chosen. Higher values of C results in a decision boundary with a smaller margin in which SVM tries to minimize the misclassified samples. According to [55] a large value of C assigns higher penalties to errors that result in training the model to minimize error with lower generalization. Additionally, when the value of C goes to 0, the model would be less complex, and the result would tolerate a large number of errors. Accordingly, higher generalization ability is achievable by choosing a lower value of C.

4.5.3 Support Vector Regression (SVR)

Unlike the SVM that was first introduced for classification purposes, SVR is a regression version of SVM for training data to predict a continuous output variable. The main difference lies in a threshold, and more specifically, SVR uses a threshold to minimize the loss function. According to the [16] SVR is a variant of Support Vector Machine (SVM) that works on the principle of derivative function $f(x)$, that maps patterns of the inputs $x_i \in R$ to the output label $y_i \in R$, based on a given training set to solve an optimization problem. Moreover, SVR for the output power of solar panels works based on the probabilistic model. This model utilizes a nonlinear mapping process to map the input vector into a higher dimensional feature space. Furthermore, SVR has an additional parameter called epsilon.

The value of ε determines a tolerance margin of errors where no penalty is imposed to errors or, in other words, the data points predicted within a distance ε from the actual data points. The value of ε represents the amount of error allowed per training data instance. The number of support vectors used to construct the regression function is affected by the epsilon parameter and limits error margin [56]. The bigger ε is, the fewer support vectors are selected.

In the field of weather forecasting, SVR has a good generalization ability and gained popularity among researchers because of non-linearity in meteorological data since its performance is superior in nonlinear modeling.

5 Performance Metrics

A key aspect of any machine learning algorithm is the prediction performance metric. The accuracy of the prediction is defined by the percentage of correct predictions. There is no general consensus on a set of acceptable metrics that can be used in predictions [57], [58]. However, several predefined commonly used metrics in PV output power are proposed to evaluate the model performance, including, Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Root Mean Squared Error [16] and R-squared. Performance evaluation metrics are defined differently, and we should expect the results to be different. Accordingly, the distribution of errors can be better understood by using multiple metrics.

5.1 Mean Absolute Error (MAE)

Mean Absolute Error is a metric that measure the difference between actual (Y) and predicted data ($\widehat{Y}$ given in the Eq. 15:

$$\text{MAE } = \frac{1}{n} \times \sum_{i=1}^{n} \left| Y_{\text{i}} - \widehat{Y}_{\text{i}} \right| \tag{15}$$

Where n is the number of fitted data points, Y corresponds to the actual value, and $\widehat{Y}$ is the predicted value.

MAE evaluation is based on the median of the errors. Furthermore, MAE is shown to be an unbiased estimator [60] thus, it might be unable to handle the large errors, or in other words, MAE gives the same weight to outliers. Therefore, when comparing across data with differing scales, it provides little insight into the errors.

5.2 Mean Absolute Percentage Error (MAPE)

MAPE is an evaluation method to measure the prediction accuracy of the model given by Eq. 16.

$$\text{MAPE } = \frac{100}{n} \times \sum_{i=1}^{n} \left| \frac{Y_{\text{i}} - \widehat{Y}_{\text{i}}}{Y_{\text{i}}} \right| \tag{16}$$

MAPE is similar to MAE, but MAPE represents the normalized absolute error over the data to compare the error with data of different scales. However, in the case of zero values in actual data, MAPE will take undefined values.

5.3 Mean Squared Error (MSE)

MSE is a common loss function by taking the squared difference of prediction, and actual data then averaging it out on the entire dataset given by Eq. 17:

$$\text{MSE } = \frac{1}{n} \times \sum_{i=1}^{n} \left(Y_{\text{i}} - \widehat{Y}_{\text{i}} \right)^2 \tag{17}$$

MSE works on the principle of averaging the errors that is sensitive to outliers.

Furthermore, in the data assimilation field, penalizing large errors through MSE proves to be very effective in improving model performance.

5.4 Root Mean Squared Error (RMSE)

RMSE works on the principle of averaging errors. RMSE is the standard deviation of residual errors that takes the root of MSE, which brings the unit back to the actual unit that is given by Eq. 18.

$$\text{RMSE } = \sqrt{\frac{1}{n} \times \sum_{i=1}^{n} \left(Y_{\text{i}} - \hat{Y}_{\text{i}} \right)^2} \tag{18}$$

Furthermore, RMSE is a biased estimator [60] that gives more weight to large errors to handle them, or in other words, when the bigger values need to be penalized, then RMSE is useful.

5.5 R-squared (R^2)

R-squared (R^2) is a regression method given by the Eq19 [10]:

$$R^2 = 1 - \frac{\sum_{i=1}^{N} \left(y_i - \widehat{y}_i \right)^2}{\sum_{i=1}^{N} \left(y_i - \bar{y} \right)^2} \tag{19}$$

This is a statistical measure that represents the proportion of the variance explained by an independent variable in a regression model [75] that represents how well the data fit the regression model. A prediction is more reliable if R2 is close to 1.0, and for measuring the relationship between estimated bias and final error, this is obviously an extremely simple way [81].

6 Validation and optimization

In this section, optimization and validation techniques are discussed.

6.1 Hyperparameter optimization

In machine learning, hyperparameter optimization refers to selecting the optimal set of parameters for an algorithm that gives the best prediction performance to a machine learning model [69]. It is important to set the hyperparameters correctly, and as an alternative to manually tuning, we can treat them as an optimization problem.

6.1.1 Manual Searching

Manual searching is a common method in which the values are hand-picked from an initial pool using experience-based judgement [70] [71]. Based on human judgment, manual searching is not consistent. Additionally, when an increasing number of hyperparameters are required to be optimized, then the manual optimization becomes complicated [72].

6.1.2 Grid Search

The traditional method for optimizing hyperparameters is a grid search by searching over a subset of the hyperparameter space of the training data.

The basic idea of the grid search method is to produce the best hyperparameter set that gives the highest accuracy.

Because parameter space of the machine learning algorithm may include spaces with real or unlimited values for some parameters, it is possible to specify a boundary to apply a grid search. Machine learning algorithms can have parameter spaces with real or unlimited values, so a specified boundary needs to be applied for the grid search [67].

In the grid search method, the dataset is randomly divided into training sets, and test sets by k-fold cross validation [68]. A set of the possible parameters are assigned manually in the grid search space, and a complete search is done over all possible parameters as is illustrated in Fig. 19 (A). The hyperparameter set that provides the best accuracy is considered the best.

However, the grid search algorithm is a complete brute-force that its computational time takes a long to execute [67].

6.1.3 Random Search

In this method, trial sets are randomly chosen from the initial pool, and the algorithm searches for the parameters by a predetermined sampling size as is illustrated in Fig. 19 (B). The random selection overrides the complete selection of all combinations.

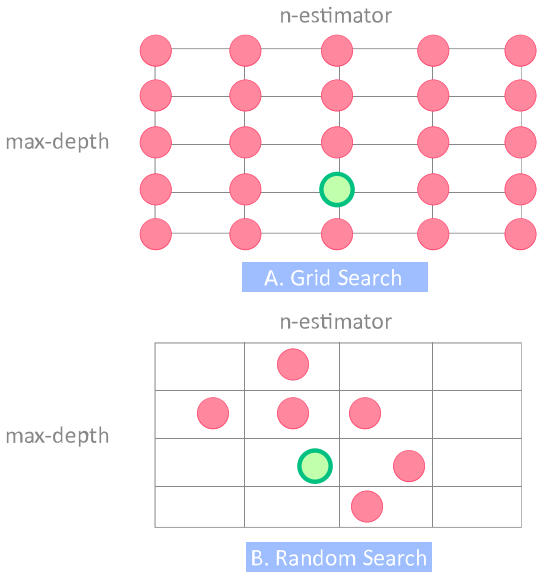

Figure 19. Hyperparameter Optimization

Therefore, random search is still limited by specified distribution over the input space [67] which often takes too long to approach the optimum.

According to [72] it is preferred to use a random-search hyperparameter optimization method over manual searches and grid searches since it gives the modeler greater control by allowing them to choose a sample size based on the available processing power. Therefore, sampling size can be varied according to the available computing resources.

6.2 Cross Validation

Cross-validation is a statistical approach used to evaluate and compare different machine learning algorithms by separating data into two segments: training a model and validating it.

K-fold cross-validation is the most basic form of cross-validation, and others methods are a variant of k-fold cross-validation. Cross-validation can be used to estimate prediction accuracy when data samples are limited. In fact, k-fold is used to estimate the skill of the model on new data.

K-fold Cross-validation involves randomly splitting the original sample into k equal-sized folds. Therefore in each iteration, a fold is held out for validation as a testing set, and the remaining folds(k-1) are utilized for training. In each iteration, the performance is evaluated by a metric. Fig. 20 illustrates the procedure for 5 fold cross-validation. As a consequence, once the performance metrics are computed for each fold, the average performance is then calculated.

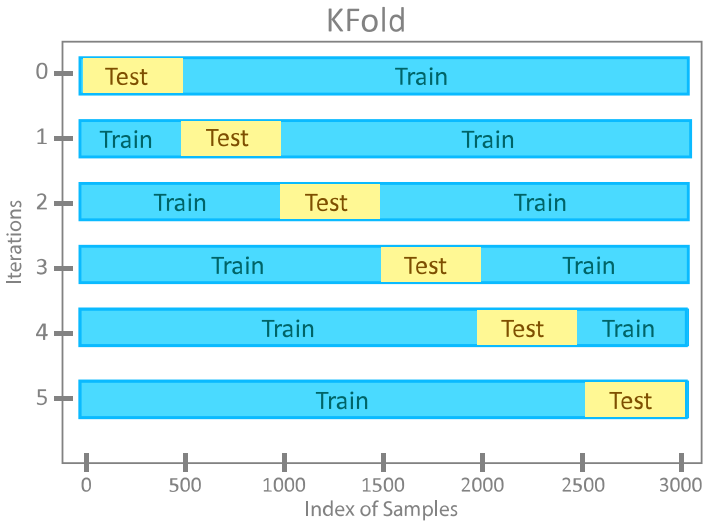

Figure 20. KFold Cross Validation

In [76] [77] [79] [78] [64], the authors have used k-fold cross validation to evaluate the model for the data corresponding to solar radiation, residential building energy consumption and short-Term photovoltaic output power predictions.

7 Results

This section presents different results from three different time intervals (30 minutes, 1 hour, and 4 hours) of our models. Moreover, empirical results based on the models for the various employed techniques are provided. Scatter plots are used to illustrate the actual data points against predicted values. Actual data points are shown along the X-axis, and predicted data points are shown along Y-axis. The red line in the scatter plot represents the perfect prediction, where the predicted value exactly matches the actual value and fits the data the most closely. Depending on the distance between a point and the ideal angle line 45 degrees, we can determine how well or poorly the prediction performed. Additionally, the actual standard deviation of the test set and predicted is given for each model. Furthermore, the number of features decreased by eight, and the final independent features are Global Horizontal Irradiance (GHI), temperature, humidity, month, and hour.

7.1 KNN

The most critical parameter for KNN is K-number. By running a grid search, we can obtain an optimal number for each value of K to estimate the accuracy. Additionally, the model requires the data to be on the same scale. Therefore, feature scaling is used for KNN.

Performance metric results for the given model are given in Fig. 21 and in the Table 8. Additionally, the standard deviation of the test set and predicted on the test set is given in Table 9.

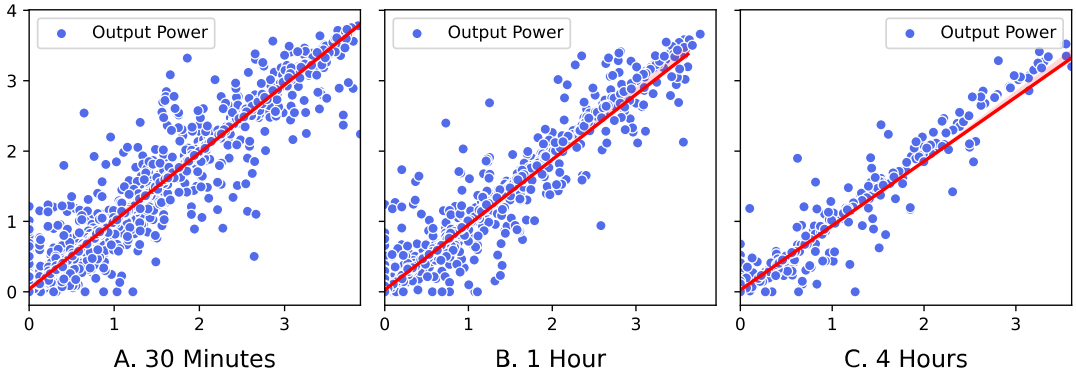

Figure 21. Predicted Output Power (Y-axis) Versus Actual Output Power (X-axis)

Table 8. Performance Metrics for KNN(kWh)

	30 Min	Hourly	4 Hours
MAPE	12.78	21.45	38.45
MAE	0.065	0.090	0.161
RMSE	0.181	0.206	0.281
R2	0.965	0.955	0.910

Table 9. Standard Deviation for Actual and Predicted of KNN

	30 Min	Hourly	4 Hours
Test Set	0.972	0.976	0.943
Predicted	0.963	0.957	0.911

7.2 XGBoost

Results obtained from XGBoost show that the model has performed well on test data, as illustrated in Fig. 22. Additionally, performance metrics are given in Table 10 and the standard deviation of the test set and predicted on the test set is given in Table 11.

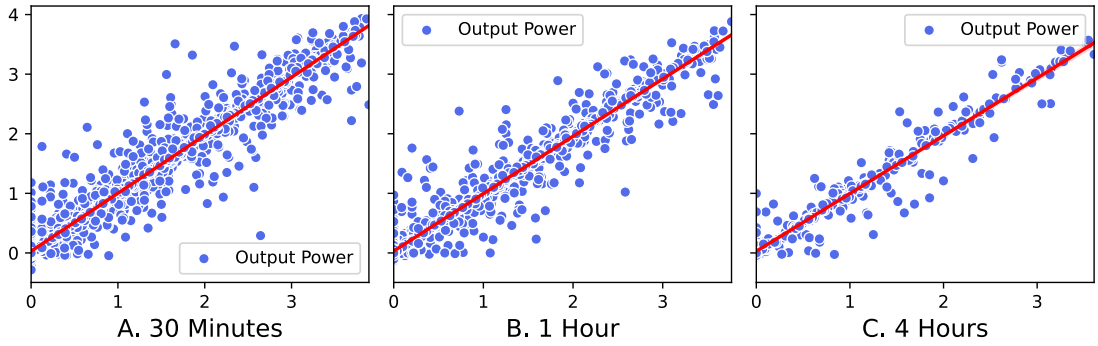

Figure 22. Predicted Output Power (Y-axis) Versus Actual Output Power (X-axis)

Table 10. Performance Metrics for XGBoost(kWh)

	30 Min	Hourly	4 Hours
MAPE	10.92	21.82	26.2
MAE	0.069	0.086	0.118
RMSE	0.179	0.182	0.214
R2	0.965	0.967	0.950

Table 11. Standard Deviation for Actual and Predicted of XGBoost

	30 Min	Hourly	4 Hours
Test Set	0.972	0.976	0.943
Predicted	0.963	0.953	0.921

7.3 MLP

MLP utilizes four hidden layer and the used activation function is Relu. Predicted output power versus actual output power is illustrated in Fig. 23. Additionally, performance metrics are given in Table 12 and standard deviation of test set and predicted on the test set is given in Table 13.

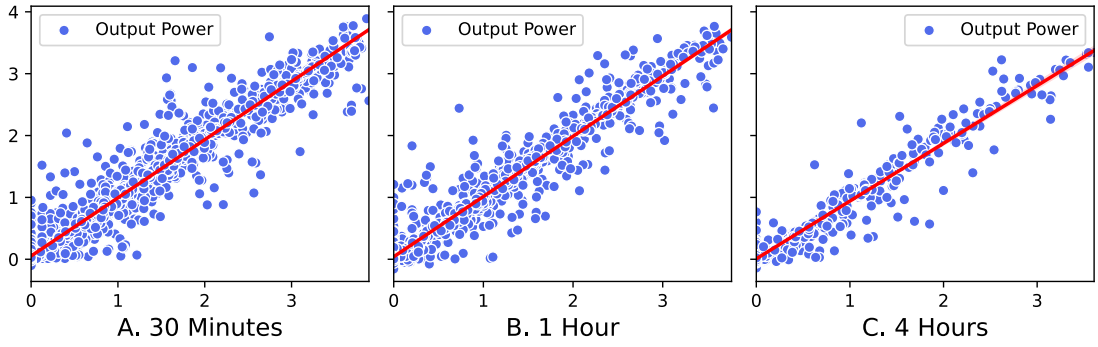

Figure 23. Predicted Output Power (Y-axis) Versus Actual Output Power (X-axis)

Table 12. Performance Metrics for MLP(kWh)

	30 Min	Hourly	4 Hours
MAPE	17.01	28.90	33.43
MAE	0.101	0.130	0.172
RMSE	0.181	0.218	0.263
R2	0.964	0.950	0.922

Table 13. Standard Deviation for Actual and Predicted of MLP

	30 Min	Hourly	4 Hours
Test Set	0.972	0.976	0.943
Predicted	0.969	1.02	0.992

7.4 SVR

Predicted output power versus actual output power is illustrated in Fig. 24. Additionally, performance metrics are given in Table 14 and standard deviation of test set and predicted on the test set is given in Table 15. For the SVR model, RBF is the main used kernel.

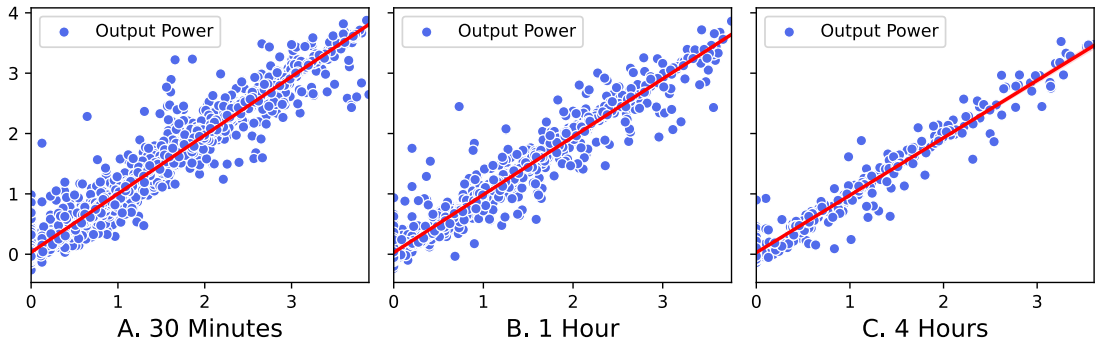

Figure 24. Predicted Output Power (Y-axis) Versus Actual Output Power (X-axis)

Table 14. Performance Metrics for SVR(kWh)

	30 Min	Hourly	4 Hours
MAPE	18.72	29.84	32.64
MAE	0.124	0.136	0.179
RMSE	0.217	0.225	0.255
R2	0.950	0.946	0.925

Table 15. Standard Deviation for Actual and Predicted of SVR

	30 Min	Hourly	4 Hours
Test Set	0.972	0.976	0.943
Predicted	0.953	0.949	0.901

7.5 Random Forest

Predicted output power versus actual output power is illustrated in Fig. 26. Additionally, performance metrics are given in Table 16 and the standard deviation of the test set and predicted on the test set is given in Table 17.

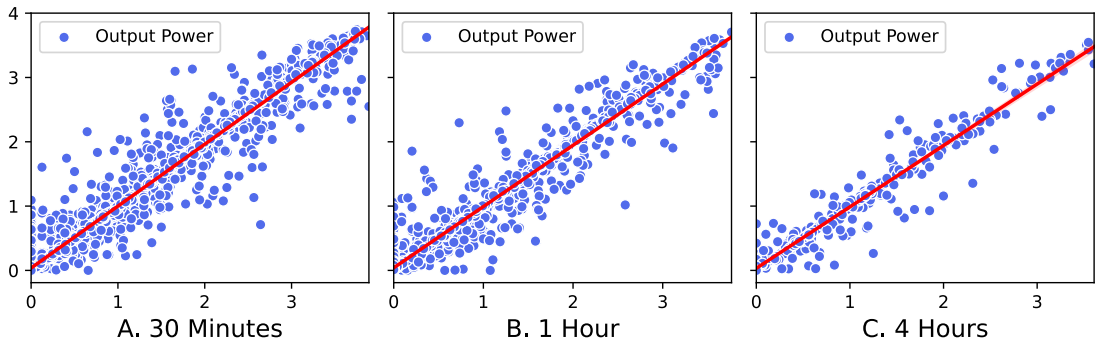

Figure 25. Predicted Output Power (Y-axis) Versus Actual Output Power (X-axis)

Table 16. Performance Metrics for RF(kWh)

	30 Min	Hourly	4 Hours
MAPE	11.30	22.00	26.62
MAE	0.069	0.089	0.127
RMSE	0.181	0.190	0.217
R2	0.965	0.961	0.946

Table 17. Standard Deviation for Actual and Predicted of RF

	30 Min	Hourly	4 Hours
Test Set	0.972	0.976	0.955
Predicted	0.951	0.939	0.908

7.6 Average of Cross-Validation Scores

Cross-validation scores for each model are given in this section. Ten-fold cross-validation is performed for each model, and scores are given in Fig. 5. Moreover, the average result of each model is given in Table 18.

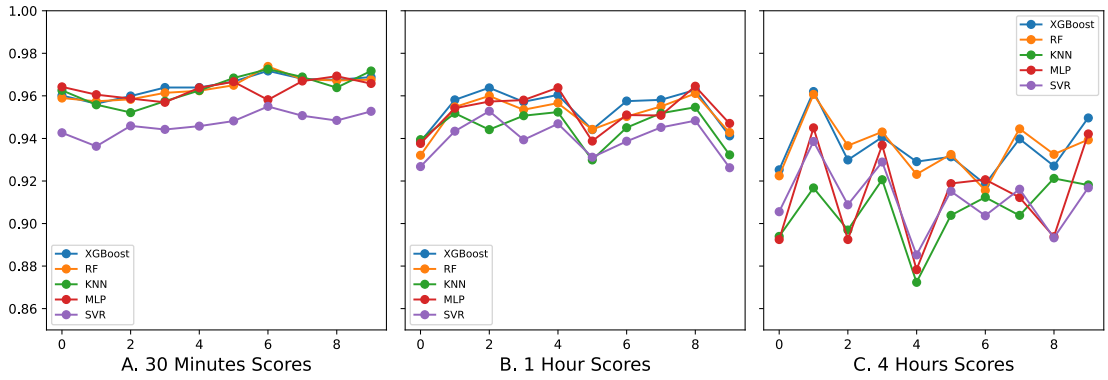

Figure 26. Cross Validation Scores - Number of Iterations(X-axis) and Scores(Y-axis)

Table 18. Average of Cross validation Scores

	30 Min	Hourly	4 Hours
KNN	0.963	0.945	0.905
XGBoost	0.964	0.954	0.935
RF	0.964	0.951	0.935
MLP	0.963	0.952	0.920
SVR	0.947	0.939	0.913

8 Conclusion

Five different machine learning models are utilized to estimate the forecasting accuracy of three different timeframes, including 30 minutes, 1 hour, and 4 hours. On the one hand, RF and XGboost are very similar in all timeframes regarding different used performance metrics. On the other hand, results obtained by MLP, SVR, and KNN show a similar pattern in their performance metrics. However, as the timeframe increase, KNN shows slightly more reduction in the performance metrics. Moreover, tree-based models show relatively more stability in their performance metrics compared to other models. Theoretically, XGBoost and RF as tree-based models should have an advantage in incorporating complex weather and solar energy data structures. Additionally, To improve the results, there are certain aspects to consider regarding methodology. One approach is implementing different machine learning models for different seasons that would probably improve the results. Splitting the year into two different weather conditions would likely improve the results. Furthermore, we have another method to improve our results as part of our future study. In practice, we added two prior output power as the derived features that significantly improved our results. To set things in perspective, the R-squared of all models was better than in the work of Tserenpurev et al. [10]. Additionally, When comparing the hourly prediction results of the SVR with the study of Ruby Nageema [13] then our model shows a good reduction (from 36 to 32.64) based on the given MAPE.

Acknowledgements

I would like to give my warmest thanks to my supervisor Dr. Chinmaya Kumar Dehury who made this thesis possible. His guidance and advice carried me through all the stages of writing and implementing this project. I would also like to give special thanks to the University of Tartu that provided this greate opportunity for me to pursue my study and I will never forget.

References

[1] Sharma, N., Sharma, P., Irwin, D., & Shenoy, P. (2011, October). Predicting solar generation from weather forecasts using machine learning. In 2011 IEEE international conference on smart grid communications (SmartGridComm) (pp. 528-533). IEEE.

[2] Jawaid, F., & NazirJunejo, K. (2016, August). Predicting daily mean solar power using machine learning regression techniques. In 2016 Sixth International Conference on Innovative Computing Technology (INTECH) (pp. 355-360). IEEE.

[3] Rodríguez, F., Fleetwood, A., Galarza, A., & Fontán, L. (2018). Predicting solar energy generation through artificial neural networks using weather forecasts for microgrid control. Renewable Energy, 126, 855-864.

[4] H. Gohar Ali, R. Vilanova Arbos, J. Herrera, A. Tobón, and J. Peláez- Restrepo, "Non-linear sliding mode controller for photovoltaic panels with maximum power point tracking," Processes, vol. 8, no. 1, p. 108, Jan. 2020.

[5] Sow, A., Mehrtash, M., Rousse, D. R., & Haillot, D. (2019). Economic analysis of residential solar photovoltaic electricity production in Canada. Sustainable Energy Technologies and Assessments, 33, 83-94.

[6] Al-Dahidi, S., Louzazni, M., & Omran, N. (2020). A local training strategy-based artificial neural network for predicting the power production of solar photovoltaic systems. IEEE Access, 8, 150262-150281.

[7] Monteiro, C., Fernandez-Jimenez, L. A., Ramirez-Rosado, I. J., Muñoz-Jimenez, A., & Lara-Santillan, P. M. (2013). Short-term forecasting models for photovoltaic plants: Analytical versus soft-computing techniques. Mathematical problems in engineering, 2013.

[8] Rich H. Inman, Hugo T.C. Pedro, and Carlos F.M. Coimbra. Solar forecasting methods for renewable energy integration. Progress in Energy and Combustion Science, 39(6):535 – 576, 2013.

[9] Antonanzas, J., Osorio, N., Escobar, R., Urraca, R., Martinez-de-Pison, F. J., & Antonanzas-Torres, F. (2016). Review of photovoltaic power forecasting. Solar energy, 136, 78-111.

[10] Chuluunsaikhan, T., Nasridinov, A., Choi, W. S., Choi, D. B., Choi, S. H., & Kim, Y. M. (2021). Predicting the Power Output of Solar Panels based on Weather and Air Pollution Features using Machine Learning. Journal of Korea Multimedia Society, 24(2), 222-232.

[11] Barrera, J. M., Reina, A., Maté, A., & Trujillo, J. C. (2020). Solar Energy Prediction Model Based on Artificial Neural Networks and Open Data. Sustainability, 12(17), 6915.

[12] Saberian, A., Hizam, H., Radzi, M. A. M., Ab Kadir, M. Z. A., & Mirzaei, M. (2014). Modelling and prediction of photovoltaic power output using artificial neural networks. International journal of Photoenergy, 2014.

[13] Nageem, R., & Jayabarathi, R. (2017). Predicting the power output of a grid-connected solar panel using multi-input support vector regression. Procedia computer science, 115, 723-730.

[14] De Leone, R., Pietrini, M., & Giovannelli, A. (2015). Photovoltaic energy production forecast using support vector regression. Neural Computing and Applications, 26(8), 1955-1962.

[15] Shi, J., Lee, W. J., Liu, Y., Yang, Y., & Wang, P. (2012). Forecasting power output of photovoltaic systems based on weather classification and support vector machines. IEEE Transactions on Industry Applications, 48(3), 1064-1069.

[16] Theocharides, S., Makrides, G., Georghiou, G. E., & Kyprianou, A. (2018, June). Machine learning algorithms for photovoltaic system power output prediction. In 2018 IEEE International Energy Conference (ENERGYCON) (pp. 1-6). IEEE.

[17] Isaksson, E., & Karpe Conde, M. (2018). Solar power forecasting with machine learning techniques.

[18] Khademi, M., Moadel, M., & Khosravi, A. (2016). Power prediction and technoeconomic analysis of a solar PV power plant by MLP-ABC and COMFAR III, considering cloudy weather conditions. International Journal of Chemical Engineering, 2016.

[19] G. Maps. Delta Center - University of Tartu, Al Arab St, Amman. Accessed: December 11, 2021. [Online]. Available: https://goo.gl/maps/1hYAXXVQSaqT7p4c6

[20] Malik, K., Bhatti, B. A., & Kamran, F. (2016, January). An approach to predict output of PV panels using weather corrected global irradiance. In 2016 International Conference on Intelligent Systems Engineering (ICISE) (pp. 111-117). IEEE.

[21] Galli, S. (n.d.). Python Feature Engineering Cookbook. O'Reilly Online Learning. Retrieved December 13, 2021, from https://www.oreilly.com/library/view/python-feature-engineering/9781789806311/ca0bc515-85c3-4b95-b55f-e6bfa2451b80.xhtml

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

[23] Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World by Valiant, Leslie (2014) Paperback (Reprint ed.). (2021). Basic Books.

[24] Kuhn, M., & Johnson, K. (2018). Applied Predictive Modeling. Springer Publishing.

[25] Bentéjac, C., Csörgő, A., & Martínez-Muñoz, G. (2021). A comparative analysis of gradient boosting algorithms. Artificial Intelligence Review, 54(3), 1937-1967.

[26] Leo Breiman. Random forests. Machine Learning, 45(1):5–32, 2001.

[27] Feng, Z., Xiong, H., Song, C., Yang, S., Zhao, B., Wang, L., ... & Huan, J. (2019, December). Securegbm: Secure multi-party gradient boosting. In 2019 IEEE International Conference on Big Data (Big Data) (pp. 1312-1321). IEEE.

[28] Sagi, O., & Rokach, L. (2018). Ensemble learning: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(4), e1249.

[29] Stearns, B., Rangel, F. M., Rangel, F., de Faria, F. F., Oliveira, J., & Ramos, A. A. D. S. (2017). Scholar Performance Prediction using Boosted Regression Trees Techniques. In ESANN.

[30] Friedman, J. H. (2002). Stochastic gradient boosting. Computational statistics & data analysis, 38(4), 367-378.

[31] Yin, D., Li, J., & Wu, G. (2021). Solving the Data Sparsity Problem in Predicting the Success of the Startups with Machine Learning Methods. arXiv preprint arXiv:2112.07985.

[32] A. Paul, D. P. Mukherjee, P. Das, A. Gangopadhyay, A. R. Chintha and S. Kundu, "Improved Random Forest for Classification," in IEEE Transactions on Image Processing, vol. 27, no. 8, pp. 4012-4024, Aug. 2018, doi: 10.1109/TIP.2018.2834830.

[33] Ho, T. K. (1995). Random decision forests. In Document analysisand recognition, 1995, Proceedings of the third interna-tional conference on (Vol. 1, pp. 278–282). IEEE.

[34] Amit Y., & Geman, D. (1997). Shape quantization and recog-nition with randomized trees. Neural Computation,9(7),1545–1588.

[35] Xhemali, D., J HINDE, C., & G STONE, R. (2009). Naïve bayes vs. decision trees vs. neural networks in the classification of training web pages. D. XHEMALI, CJ HINDE and Roger G. STONE," Naive Bayes vs. Decision Trees vs. Neural Networks in the Classification of Training Web Pages", International Journal of Computer Science Issues, IJCSI, Volume 4, Issue 1, pp16-23, September 2009, 4(1).

[36] Liaw A, Wiener M (2002) Classification and regression by random forest. R News 2(3):18–22

[37] Biau, G., & Scornet, E. (2016). A random forest guided tour. Test, 25(2), 197-227.

[38] Yen, C. F., Hsieh, H. Y., Su, K. W., Yu, M. C., & Leu, J. S. (2018). Solar Power Prediction via Support Vector Machine and Random Forest. In E3S Web of Conferences (Vol. 69, p. 01004). EDP Sciences.

[39] Ramli, N. A., Hamid, M. F. A., Azhan, N. H., & Ishak, M. A. A. S. (2019, July). Solar power generation prediction by using k-nearest neighbor method. In AIP Conference Proceedings (Vol. 2129, No. 1, p. 020116). AIP Publishing LLC.

[40] Gao, X., & Li, G. (2020). A KNN model based on manhattan distance to identify the SNARE proteins. IEEE Access, 8, 112922-112931.

[41] T. Hastie J. Gareth, D. Witten and R. Tibshirani. An Introduction to Statistical Learning with Applications in R. Springer texts in statistics An introduction to statistical learning. Springer, 2013.

[42] Aggarwal, C. C., Hinneburg, A., & Keim, D. A. (2001, January). On the surprising behavior of distance metrics in high dimensional space. In International conference on database theory (pp. 420-434). Springer, Berlin, Heidelberg.

[43] Noriega, L. (2005). Multilayer perceptron tutorial. School of Computing. Staffordshire University.

[44] Hecht-Nielsen, Theory of the backpropagation neural network. In: International 2989 joint conference on neural networks, Washington, DC, USA. 1989.

[45] Aslam, M., Lee, J. M., & Hong, S. (2020). A multi-layer perceptron based deep learning model to quantify the energy potentials of a thin film a-Si PV system. Energy Reports, 6, 1331-1336.

[46] Parvez, I., Sarwat, A., Debnath, A., Olowu, T., Dastgir, M. G., & Riggs, H. (2020, March). Multi-layer perceptron based photovoltaic forecasting for rooftop pv applications in smart grid. In 2020 SoutheastCon (pp. 1-6). IEEE.

[47] S.A. Kalogirou, Artificial neural networks in renewable energy systems applications: a review, Renew. Sustain. Energy Rev. 5 (2001) 373e401.

[48] Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. Neural computation, 4(1), 1-58.

[49] Karthikeyan, A., Mirza, S., Hur, B., Pearlstein, G., & Ledbetter, R. (2020). Quantifying Variable Importance in Predicting Critical Span Length and Scour Depth for Failure of Onshore River Crossing Pipelines Using ANN. Journal of Marine Science and Engineering, 8(11), 840.

[50] Doan, C. D., & Liong, S. Y. (2004, July). Generalization for multilayer neural network bayesian regularization or early stopping. In Proceedings of Asia Pacific association of hydrology and water resources 2nd conference (pp. 5-8).

[51] Montavon, G., Orr, G., & Müller, K. R. (Eds.). (2012). Neural networks: tricks of the trade (Vol. 7700). springer.

[52] Attoh-Okine, N. O. (1999). Analysis of learning rate and momentum term in backpropagation neural network algorithm trained to predict pavement performance. Advances in engineering software, 30(4), 291-302.

[53] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

[54] Durgesh, K. S., & Lekha, B. (2010). Data classification using support vector machine. Journal of theoretical and applied information technology, 12(1), 1-7.

[55] Wu, C. H., Ho, J. M., & Lee, D. T. (2004). Travel-time prediction with support vector regression. IEEE transactions on intelligent transportation systems, 5(4), 276-281.

[56] Çelikyılmaz, A., & Türkşen, İ. B. (2009). Modeling Uncertainty with Fuzzy Logic With Recent Theory and Applications Introduction. Modeling Uncertainty With Fuzzy Logic: With Recent Theory And Applications.

[57] Dammers, E. (1993). Measurement in the ex post evaluation of forecasts. Quality and Quantity, 27(1), 31-45.

[58] Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. International journal of forecasting, 22(4), 679-688.

[59] Tao Hong, Pierre Pinson, Shu Fan, Hamidreza Zareipour, Alberto Troccoli, and Rob J. Hyndman. Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond. International Journal of Forecasting, 32(3):896 – 913, 2016.

[60] Brassington, G. (2017, April). Mean absolute error and root mean square error: which is the better metric for assessing model performance?. In EGU General Assembly Conference Abstracts (p. 3574).

[61] V. Barnett and T. Lewis. Outliers in Statistical Data. John Wiley and Sons, 1978

[62] Escalante, H. J. (2005). A comparison of outlier detection algorithms for machine learning. In Proceedings of the International Conference on Communications in Computing (pp. 228-237).

[63] A. Arning, R. Agrawal, and P. Raghavan. A linear method for deviation detection in large databases. In KDDM, pages 164–169, 1996

[64] Zhu, R., Guo, W., & Gong, X. (2019). Short-term photovoltaic power output prediction based on k-fold cross-validation and an ensemble model. Energies, 12(7), 1220.

[65] Zhou, H.; Deng, Z.; Xia, Y.; Fu, M. A new sampling method in particle filter based on pearson correlation coefficient. Neurocomputing 2016, 216, 208–215.

[66] Kim, N.; Park, S.; Lee, J.; Choi, K.J. Load profile extraction by mean-shift clustering with sample pearson correlation coefficient distance. Energies 2018, 11, 2397.

[67] Liashchynskyi, P., & Liashchynskyi, P. (2019). Grid search, random search, genetic algorithm: a big comparison for NAS. arXiv preprint arXiv:1912.06059.

[68] Burman, P. (1989). A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. Biometrika, 76(3), 503-514.

[69] Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. Advances in neural information processing systems, 24.

[70] Li, Q.; Meng, Q.; Cai, J.; Yoshino, H.; Mochida, A. Predicting hourly cooling load in the building: A comparison of support vector machine and different artificial neural networks. Energy Convers. Manag. 2009, 50, 90–96.

[71] Massana, J.; Pous, C.; Burgas, L.; Melendez, J.; Colomer, J. Short-term load forecasting in a non-residential building contrasting models and attributes. Energy Build. 2015, 92, 322–330

[72] Krishnadas, G., & Kiprakis, A. (2020). A machine learning pipeline for demand response capacity scheduling. Energies, 13(7), 1848.

[73] Zhao, Y., Lehman, B., Ball, R., Mosesian, J., & de Palma, J. F. (2013, March). Outlier detection rules for fault detection in solar photovoltaic arrays. In 2013 Twenty-Eighth Annual IEEE Applied Power Electronics Conference and Exposition (APEC) (pp. 2913-2920). IEEE.

[74] Mohamad, I. B., & Usman, D. (2013). Standardization and its effects on K-means clustering algorithm. Research Journal of Applied Sciences, Engineering and Technology, 6(17), 3299-3303.

[75] Manohar, H. L. (2019). The mediating effect of coopetition between the process of social alliance building and social innovation.

[76] Saud, S., Jamil, B., Upadhyay, Y., & Irshad, K. (2020). Performance improvement of empirical models for estimation of global solar radiation in India: A k-fold cross-validation approach. Sustainable Energy Technologies and Assessments, 40, 100768.

[77] Rohani, A., Taki, M., & Abdollahpour, M. (2018). A novel soft computing model (Gaussian process regression with K-fold cross validation) for daily and monthly solar radiation forecasting (Part: I). Renewable Energy, 115, 411-422.

[78] Tabrizchi, H., Javidi, M. M., & Amirzadeh, V. (2021). Estimates of residential building energy consumption using a multi-verse optimizer-based support vector machine with k-fold cross-validation. Evolving Systems, 12(3), 755-767.

[79] Taki, M., Rohani, A., & Yildizhan, H. (2021). Application of machine learning for solar radiation modeling. Theoretical and Applied Climatology, 143(3), 1599-1613.

[80] Fürnkranz J. (2011) Decision Tree. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_204

[81] Yin, H., Yao, X., Tino, P., Corchado, E., & Byrne, W. (Eds.). (2007). Intelligent Data Engineering and Automated Learning-IDEAL 2007: 8th International Conference, Birmingham, UK, December 16-19, 2007, Proceedings (Vol. 4881). Springer.

[82] Labani, M., Moradi, P., Ahmadizar, F., & Jalili, M. (2018). A novel multivariate filter method for feature selection in text classification problems. Engineering Applications of Artificial Intelligence, 70, 25-37.

[83] Belkoura, S., Zanin, M., & LaTorre, A. (2019). Fostering interpretability of data mining models through data perturbation. Expert Systems with Applications, 137, 191-201.

[84] Heymann, S., Latapy, M., & Magnien, C. (2012, August). Outskewer: Using skewness to spot outliers in samples and time series. In 2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (pp. 527-534). IEEE.

[85] Huang, R., McIntyre, S., Song, M., & Ou, Z. (2018). An attention-based recommender system to predict contextual intent based on choice histories across and within sessions. Applied Sciences, 8(12), 2426.

[86] Muthukrishnan, R., & Rohini, R. (2016, October). LASSO: A feature selection technique in predictive modeling for machine learning. In 2016 IEEE international conference on advances in computer applications (ICACA) (pp. 18-20). IEEE.

[87] Li, F., Yang, Y., & Xing, E. (2005). From lasso regression to feature vector machine. Advances in neural information processing systems, 18.

[88] Hongjun, S., Hiroyuki, F., & Masato, N. A. K. A. J. I. M. A. (2019). Variation Of Density Functions For The Distribution Of Residuals Between GMPE And Observation Data Based On The NGA-W2 Database.

[89] Riigi Ilmateenistus |. (2021, October 9). Riigi Ilmateenistus. https://www.ilmateenistus.ee/

[90] University Of Tartu. (2021, October 1). Institutional Development Service | Välisveeb. University of Tartu. https://omi.ut.ee/en/institutional-development-service

[91] Hill, A. J., Herman, G. R., & Schumacher, R. S. (2020). Forecasting severe weather with random forests. Monthly Weather Review, 148(5), 2135-2161.

Appendix

I. Acronyms

ANN Artificial Neural Network

AR Adaptive Recursive Linear

FFBP Feed-forward Back Propagation

GBRT Gradient Boosted Regression Trees

GHI Global Horizontal Irradiance

GRNN General Regression Neural Network

IQR Interquartile Range

KNN K-Nearest Neighbour

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

MLP Multilayer Perceptron

NWP Numerical Weather Prediction

PV photovoltaic

RBF Radial Basis Function

RE Renewable Energy

RF Random Forest

RMSE Root Mean Square Error

ROI Return on Investment

RT Regression Trees

SGD Stochastic Gradient Descent

SVM Support Vector Machine

SVR Support Vector Regression

XGBoost Extreme Gradient Boosting

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

Mehdi Hatamian,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Predicting location-based green energy availability in smart buildings,

   supervised by Dr. Chinmaya Kumar Dehury.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Mehdi Hatamian
09/03/2022