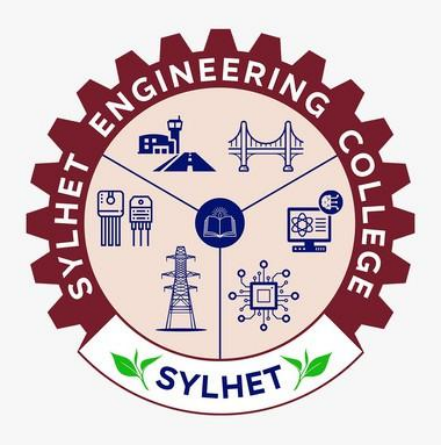
A Thesis Submitted to the Sylhet Engineering College for the Degree of
Bachelor of Science in Electrical and Electronic Engineering

A Meta Model Approach and Comparative Analysis With Conventional Models for Accurate Solar Energy Prediction for Sylhet Division of Bangladesh Using Meteorological Statistics

By
Amlan Sarker Arup
&
Amit Kanti Ray Badhon

Supervised by
Md. Ashraful Alam
Lecturer
Department of Electrical and Electronic Engineering
Sylhet Engineering College



June, 2025
Sylhet Engineering College, Sylhet
Affiliated with
Shahjalal University of Science & Technology (SUST)

The thesis titled **"A Meta Model Approach and Comparative Analysis With Conventional Models for Accurate Solar Power Prediction for Sylhet Division of Bangladesh Using Meteorological Statistics"** submitted by **Amit Kanti Ray Badhon** and **Amlan Sarkar Arup** ; Student ID: 2019338503, 2019338519; Session 2019-2020, to the Department of Electrical and Electronic Engineering, Sylhet Engineering College, has been accepted as satisfactory in partial fulfilment of the requirement for the Degree of Bachelor of Science in Electrical and Electronic Engineering and approved as to its style and contents.

BOARD OF EXAMINERS

Md. Shahid Iqbal
Assistant Professor and Head
Department of Electrical and Electronic Engineering
Sylhet Engineering College, Sylhet.

Chairman

Salman Fazle Rabby
Assistant Professor
Department of Electrical and Electronic Engineering
Sylhet Engineering College, Sylhet.

Member

Apurba Biswas
Assistant Professor
Department of Electrical and Electronic Engineering
Sylhet Engineering College, Sylhet.

Member

Md. Ashraful Alam
Lecturer
Department of Electrical and Electronic Engineering
Sylhet Engineering College, Sylhet.

Supervisor & Member

Mahedi Kamal Ahmed
Lecturer,
Department of Electrical and Electronic Engineering
Sylhet Engineering College, Sylhet.

Member

Arif Ahammad
Assistant Professor
Department of Electrical and Electronic Shahjalal University of Science & Technology, Sylhet.

Member (External)

Acknowledgement

We begin by extending our gratitude to the Almighty, whose guidance and sustenance have enabled us to embark on this journey and complete the requirements for the Bachelor of Science in Electrical and Electronics Engineering (EEE), including our thesis work. It is through His grace that we have attained the strength and determination to achieve this milestone. We want to express our sincere appreciation to our esteemed supervisor, **Md. Ashraful Alam**, Lecturer of the Department of Electrical and Electronics Engineering at Sylhet Engineering College. His invaluable academic mentorship, exemplary leadership, positive encouragement, insightful advice, and unwavering support have been instrumental throughout our educational pursuit. We are truly grateful for his guidance. Furthermore, we extend our thanks to our other faculty members, **Md. Shahid Iqbal, Salman Fazle Rabby**, for their kindness, support, and cooperation during our academic journey. Their assistance has been greatly appreciated. Our heartfelt gratitude also goes to our parents, siblings, classmates, and friends in the EEE department at SEC. We acknowledge the sacrifices, prayers, encouragement, and unwavering support they have provided, which have contributed significantly to our success. We are indebted to them for their role in shaping our academic and personal development.

Abstract

This study proposes a comprehensive machine learning-based approach for predicting solar energy generation in Sylhet, Bangladesh, by leveraging historical solar output data combined with key meteorological parameters. The primary objectives are to identify the most accurate machine learning model for photovoltaic (PV) system forecasting and to improve prediction accuracy through the integration of weather-related features. The dataset, obtained from Visual Crossing, comprises daily average records from 2010 to 2024, including variables such as temperature, humidity, wind direction, solar radiation, and more. To ensure the integrity of the dataset, a series of pre-processing steps were applied: missing values were treated using mean and median imputation depending on the proportion of missingness; outliers were identified and removed using the Interquartile Range (IQR) method; and highly correlated features were dropped based on a correlation matrix. Several machine learning models—ANN, KNN, SVR, Decision Tree, Random Forest, and XG-Boost—were trained and evaluated. Evaluation metrics, including Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and the coefficient of determination (R^2), were used to assess model performance. The meta-model demonstrated superior results across all metrics, achieving the lowest MAE (0.0979), RMSE (0.3803), MAPE (0.0090), and the highest R^2 (0.9936), indicating excellent predictive capability. These findings highlight the effectiveness of ensemble meta-modelling for reliable and accurate solar energy forecasting in renewable energy application.

Keywords*: Solar energy prediction, Meta-model, Machine learning, PV systems, IQR method, Outlier handling, Sylhet, Renewable energy forecasting, Meteorological data, Visual Crossing.*

Table of Contents

Chapter 1: Introduction

Photovoltaic (PV) systems can be a viable alternative to mitigate the electrical energy crisis in developing countries like Bangladesh. With the rapid depletion of non-renewable energy sources and increasing demand for electricity, solar energy emerges as a sustainable and environmentally friendly solution. However, integrating solar energy into existing power grids has technical challenges, primarily due to its irregular nature influenced by variable environmental and meteorological factors. As a result, accurate prediction of solar energy generation becomes essential for ensuring grid stability and efficient energy management. This research specifically addresses this challenge by focusing on the applications of machine learning techniques to enhance the precision of solar energy prediction, leveraging comprehensive meteorological data. Our approach involves developing a meta-model that combines the strengths of several advanced machine learning algorithms and doing a comparative analysis with conventional machine learning models including: Support Vector Regression (SVR), Linear Regression (LR), Random Forest Regression (RFR), K-Nearest Neighbors (KNN), Artificial Neural Network (ANN), Long Short-Term Memory (LSTM) networks, and XGBOOST. The final output is achieved by a Linear Regression (LR) layer, which acts as the ultimate predictor to deliver robust and accurate solar energy forecasts.

1.1 What is solar energy?

Solar energy is the energy harnessed from the sun through solar panels, which convert sunlight into electrical power. The process involves absorbing solar irradiance using photovoltaic systems and converting it into usable electricity. Several environmental factors, such as temperature, humidity, wind speed and direction, sea-level pressure, and solar irradiance, significantly affect solar energy production. It is renewable, widely available, and produces zero emissions, making it a critical component in transitioning to sustainable energy sources.

1.2 Why Energy Prediction is Necessary?

Although solar energy presents numerous advantages, its efficiency and reliability are heavily influenced by dynamic weather conditions. The unpredictability of solar irradiance and other atmospheric variables creates complications in maintaining grid stability when

integrating PV systems. Forecasting solar energy generation is essential for:

- Efficient Grid Management: Helps in balancing supply and demand.
- Load Forecasting: Assists utilities in planning and operation.
- Resource Optimization: Enhances the economic operation of power systems.
- Reducing Dependence on Non-renewable Sources: Helps shift toward greener alternatives.

Without precise prediction models, energy planners and grid operators face difficulties in utilizing solar energy effectively, leading to possible energy surpluses or losses.

1.3 Key Functions of Solar Energy Estimation

Solar energy estimation serves multiple vital functions in modern energy systems:

•Forecasting Power Generation

o Helps to estimate the amount of energy that will be produced under given weather conditions.

•Grid Load Balancing

o Maintains grid equilibrium by aligning generation with demand.

•Energy Storage Planning

o Assists in determining storage capacity needs during surplus and deficit periods.

•Operational Scheduling

o Enables efficient scheduling of power generation assets.

•Carbon Emission Reduction

o Supports sustainable practices by reducing dependence on fossil fuels.

These functions are critical for the stable and sustainable operation of power systems using solar energy.

1.4 Impact on Grid Stability

The integration of PV systems into the national or regional power grid can lead to instability due to the fluctuating nature of solar energy. Changes in solar irradiance caused by cloud cover, temperature shifts, or humidity variations lead to unpredictable energy outputs. This unpredictability can affect grid reliability, energy delivery, and load management. Therefore, accurate estimation of solar energy production plays a vital role in ensuring the stability and reliability of the grid. Improved prediction models contribute to better energy management and reduce the risks of power shortages or overloads.

1.5 Statement of Problems

Despite the potential of solar energy, several challenges hinder its effective use:

- Conventional statistical models are insufficient in capturing complex, non-linear relationships between environmental variables and energy output.
- Manual forecasting is inefficient due to the complicated dependencies of meteorological features.
- Uncertainty in solar energy production affects grid reliability and planning.
- Need for advanced models that can adapt to dynamic weather patterns and predict energy output with high accuracy.

These challenges necessitate the use of machine learning techniques to enhance the reliability and accuracy of solar energy forecasting.

1.6 Purpose of Research

The primary aim of this research is to develop and evaluate various machine learning models to predict solar energy generation more accurately. By utilizing historical weather data and PV system outputs, this study seeks to identify the most efficient algorithm for forecasting solar power. The research emphasizes adaptability, precision, and real-world applicability of the developed models.

1.7 Objectives

This research is conducted with the following specific objectives:

1. To identify the most accurate machine learning model for solar energy prediction in PV systems.
2. To increase the prediction accuracy by integrating historical solar energy data with meteorological features.
3. To explore the performance of various machine learning algorithms, including:

- Linear Regression (LR)
- Random Forest Regression (RFR)
- XGBOOST
- Support Vector Regression (SVR)
- K-Nearest Neighbors (KNN)
- Artificial Neural Network (ANN)
- Meta-Model

1.8 Definition of Terms

- Photovoltaic (PV) System: A technology that converts solar energy into electrical energy using solar cells.
- Solar Energy: Energy derived from the sun's radiation, harnessed through solar panels.
- Solar Irradiance: The power per unit area received from the sun in the form of electromagnetic radiation.
- Forecasting: The process of predicting future values based on historical data.
- Machine Learning (ML): A subset of artificial intelligence that enables computers to learn patterns from data and make predictions.
- Non-linear Relationships: Complex interactions between variables that do not follow a straight line when plotted.
- Energy Management: The process of controlling, and conserving energy in a system

1.9 Organization of the Report

This report is organized into six chapters, each focusing on a specific aspect of the research on the estimation of solar energy using machine learning approaches.

Chapter 2: Literature Review: This chapter reviews relevant studies and existing methods in the field of solar energy forecasting, with the use of machine learning techniques like Support Vector Regression (SVR), Linear Regression (LR), Random Forest Regression (RFR), K-Nearest Neighbors (KNN), Stacking Regressor, Artificial Neural Network (ANN), Long Short-Term Memory (LSTM) networks, and Extreme Gradient Booster (XGBOOST), etc.

Chapter 3: Theoretical Analysis: This chapter provides an introduction to Support Vector Regression (SVR), Linear Regression (LR), Random Forest Regression (RFR), K- Nearest Neighbors (KNN), Artificial Neural Network (ANN), Long Short-Term Memory (LSTM) networks, and Extreme Gradient Booster (XGBOOST) , and explains the architecture of the meta model. It also includes a detailed analysis of various evaluation metrics used to assess the performance of solar energy prediction models.

Chapter 4: Methodology: Chapter 4 outlines the methodology used for this research, including data collection and processing, data splitting for training and testing, and model development. The chapter describes how the machine learning model was designed and trained for the estimation of solar energy generation.

Chapter 5: Results and Discussion: This chapter presents the results of the machine learning models developed in Chapter 4. It includes a detailed analysis of the performance of the meta model and the other approached models. The discussion evaluates the accuracy and effectiveness of the models and highlights areas of improvement.

Chapter 6: Conclusion and Future Works: Chapter 6 summarizes the key findings of the research, concludes the effectiveness of the machine learning based forecasting model, and outlines potential areas for future research and improvements in solar energy estimation.

1.10 Summary

In summary, solar energy presents an effective solution to energy shortages, particularly in developing countries. However, its integration into power grid is challenged by variability in production caused by environmental factors. This study explores several advanced ML methods to forecast solar power with high accuracy.

Chapter 2: Literature Review

A significant advancement has been noticed recently in research related to solar energy prediction. Numerous studies have been conducted to obtain accurate prediction models using machine learning techniques.

In a study in Morocco, Younes Ledmaoui et al. (2023) introduced a comparative analysis of machine learning algorithms to improve the accuracy of solar energy prediction, using six machine learning (ML) algorithms: Support Vector Regression (SVR), Artificial Neural Network (ANN), Decision Tree (DT), Random Forest (RF), Generalized Additive Model (GAM) and Extreme Gradient Boosting (XGBOOST), based on solar power plant daily data installed in Benguerir city between January and December 2022. ANN outperformed all five other models for all assessment measures based on its performance metrics such as root mean square error (RMSE), mean average error (MAE), coefficient of determination (R^2), etc.

In "Analysis of Solar Power Generation Forecasting Using Machine Learning Techniques," K. Anuradha, Deekshitha Erlapally, G. Karuna, V. Srilakshmi, and K. Adilakshmi (2021) aimed to address the critical need for accurate solar power forecasting due to the unpredictable nature of photovoltaic (PV) systems, which are highly dependent on environmental conditions like irradiance, humidity, PV surface temperature, and wind speed. Their approach involved applying various machine learning regression models, specifically Support Vector Machine Regressor, Random Forest Regressor, and Linear Regression, to predict solar power generation using historical data taken every 3 hours from the National Weather Service (NWS) as model inputs. The output of their analysis was that the Random Forest Regressor significantly outperformed the other two regression models, demonstrating superior accuracy in forecasting solar power generation.

Irfan Khan Tanoli et al. (2024) conducted a study to obtain high-performance solar radiation prediction using machine learning models by investigating the relationship between input and output parameters. This research accurately predicts surface solar radiation for 45 glaciers across Khyber Pakhtunkhwa (KPK) and Gilgit in northern Pakistan, leveraging the CERES dataset spanning from 2001 to 2021.

By analysing input parameters like date, temperature, pressure, precipitation, and aerosol, the study employed various machine learning algorithms, including Linear Regression, Decision Tree, Random Forest, Feed-Forward Neural Network (FFNN), and Long Short-Term Memory. The FFNN algorithm consistently demonstrated an R- squared (R2) score of the highest accuracy for both regions, achieving an 0.916399 for KPK and 0.886703 for Gilgit, thereby significantly contributing to renewable energy planning, climate change assessments, and glacier monitoring in the region.

In another study, titled “Hybrid KNN-SVM machine learning approach for solar power forecasting”, Nishant Saxena et al. (2024) used metrics such as Hourly Average Temperature (HAT), Hourly Total Sunlight Duration (HTSD), Hourly Total Global Solar Radiation (HTGSR), and Hourly Total Photovoltaic Energy Generation (HTPEG) to increase the reliability of the power system. This study proposes a hybrid machine learning model to enhance solar power forecasting, a critical need for renewable energy plants facing weather uncertainties. Moving beyond limitations of traditional methods like LSTM and SVM, the researchers developed a novel blend of K-Nearest Neighbors (KNN) and Support Vector Machine (SVM), leveraging their respective strengths in structural and data diversity. Validation using a real-time Jodhpur dataset from 2020, containing various hourly weather and energy generation metrics, showed the hybrid model achieving a 98% prediction accuracy. This outcome represents a significant improvement over conventional LSTM, providing a more reliable tool for power system operations and energy trading.

The research by Md Shafiul Alam et al. (2023) addresses the need for accurate solar irradiance forecasting in Bangladesh to facilitate the integration of solar PV into the utility grid, a necessity driven by increasing energy demand.The study intends to forecast solar radiation in Bangladesh using ensemble machine-learning models, examining the influence of various meteorological factors. The approach involved collecting meteorological data from 32 stations across Bangladesh, encompassing maximum temperature, minimum temperature, total rain, humidity, sunshine, wind speed, cloud coverage, and irradiance. Four ensemble machine learning algorithms were developed and tested: ADABOOST Regression (ABR), Gradient-Boosting Regression (GBR), Random Forest Regression (RFR), and Bagging Regression (BR).

The results indicated that Gradient-Boosting Regression (GBR) initially provided the best performance with default parameters, demonstrating the lowest standard deviation of errors. To further enhance prediction accuracy, the key hyperparameters of the GBR model were fine-tuned using a grid-search algorithm. The output of the optimized GBR model was outstanding, achieving the highest coefficient of determination (R2) of 0.9995 on the testing dataset. It also yielded the lowest root mean squared error (0.0007), mean absolute percentage error (0.0052), and mean squared logarithmic error (0.0001), implying superior performance. The absolute error of the prediction remained within a narrow range, confirming the model's excellent performance.

In another research, Alain K. Chaaban and Najd Alfadl (2023) tackle the challenging task of accurately forecasting solar energy yield, crucial for grid areas with high photovoltaic shares, by moving beyond the limitations of traditional statistical and physical models. They intended to compare and evaluate various machine learning models to predict PV energy yield using weather forecast data. Their approach involved developing a methodology to assess Artificial Neural Networks (ANNs), Random Forest Regression (RFR), and Long Short-Term Memory (LSTM) networks, utilizing real-world datasets from a large-scale industrial solar project that included historical PV, meteorological, and solar irradiation data. The results demonstrated that the Random Forest Algorithm consistently outperformed other models, achieving a Mean Absolute Error (MAE) of 0.06 and a Root Mean Square Error (RMSE) of 0.15 when using historical meteorological data. Furthermore, combining meteorological data with a solar irradiation dataset significantly improved accuracy, yielding an MAE of 0.03% and an RMSE of 0.09%. This high accuracy and stability, confirmed by validation analysis, means the output of their proposed methodology offers valuable information for PV system operators, grid managers, and energy planners, aiding in the optimization of solar energy resources.

In a study, Elissaios Sarmas et al. (2023) approached the problem of blending four LSTM prediction models. This research proposes a meta-learning method to enhance one-hour-ahead deterministic short-term photovoltaic (PV) power forecasts, recognizing that optimal Deep Learning (DL) architectures vary with weather and PV system specifics. The approach dynamically blends predictions from four diverse Long Short-Term Memory (LSTM) base models, crucially without using numerical weather

predictions to enhance generalizability. Evaluated on 2.5 years of hourly data from three Lisbon rooftop PV plants, the meta-learner significantly improved accuracy by up to 5% over the best single base model per plant and 4.5% over equal-weighted combinations, demonstrating statistically significant gains, especially during peak production hours, by adapting to which base model performs best under specific conditions.

In a study, conducted in Fukuoka, Japan, Bouchaib Zazoum investigated machine learning models for solar photovoltaic (PV) power prediction. The study aimed to model the relationship between PV power output and input parameters: solar PV panel temperature, ambient temperature, solar flux, time of day, and relative humidity. Comparing Support Vector Machine (SVM) and Gaussian Process Regression (GPR) algorithms, the research identified the Matern 5/2 GPR as the optimal performer, achieving an RMSE of 7.967 and MAE of 5.302. This significantly surpassed Cubic SVM, which yielded the worst results (RMSE 21.72, MAE 15.667). While other GPR variants (Rational Quadratic, Squared Exponential) also showed high accuracy (R2 of 0.98), all SVM variants (Cubic, Quadratic, Linear) consistently exhibited larger deviations. The findings conclusively support the efficacy of GPR models for solar PV power prediction and emphasize the importance of appropriate algorithm selection in modeling complex environmental-PV output relationships.

Md Jobayer et al. (2023) conducted a systematic review to address the critical need for reliable photovoltaic (PV) system performance assessment through machine learning (ML). Analysing ML-based PV parameter estimation studies from 2020-2022, they found Neural Networks (32.55%) were the most popular ML method, followed by Random Vector Functional Link (13.95%) and Support Vector Machine (9.30%). Data primarily stemmed from simulations (66%), with RMSE (29.1%), MAE (17.5%), and R^2 (15.9%) as top error metrics. Specific studies, like Theocharides et al. (finding ANN suitable over one year) and Duchaud et al. (identifying ARMA as top for short/long-term irradiance forecasting with nRMSEs of 0.8% and 1.6%, respectively), highlight diverse model applications and performance across varying durations and locations. This review aims to guide researchers in assessing ML algorithm projections for PV parameter estimation, fostering robust governmental frameworks and optimizing the PV industry.

In 2024, Halima Haque and Md. Abdur Razzak et al. conducted a study on medium-term energy demand forecasting in Bangladesh using machine learning. They analyzed nearly four years of energy consumption data from various apartment types within a sub-district of a Bangladeshi divisional city, incorporating relevant attributes. The research compared K-nearest neighbor (KNN) and Light Gradient Boosting Machine (Light-GBM) models. KNN demonstrated superior performance with 72% accuracy (MAE: 13; MSE: 49072; RMSE: 222), outperforming Light-GBM, which achieved 57% accuracy (MAE: 19; MSE: 90382; RMSE: 301). The authors emphasize that including entities related to the forecasting process notably improved these results, providing valuable insights for future energy system planning and optimization in Bangladesh.

2.1 Summary

The literature review presented in Chapter 2 highlights various methods for solar energy forecasting in large-scale power systems, focusing primarily on the machine learning approaches like K-Nearest Neighbours (KNN), Support Vector Regressor (SVR), Long Short-Term Memory (LSTM), Decision Tree (DT), Random Forest Regression (RFR), and Extreme Gradient Booster (XGBOOST) etc. These methods have shown significant promise in improving forecasting accuracy, particularly when integrated into hybrid models that leverage the strengths of both blended models.

Chapter 3: Theoretical Analysis

Chapter 3 focuses on the K-Nearest Neighbours (KNN), Support Vector Regressor (SVR), Long Short-Term Memory (LSTM), Decision Tree (DT), Random Forest Regression (RFR), and Extreme Gradient Booster (XGBOOST) models, which are essential for solar energy forecasting using meteorological data. It highlights how the models work and are made, and how they overcome the limitations. The chapter also explores a meta-model and evaluation metrics to assess forecasting accuracy in power system applications.

3.1 Introduction to K-Nearest Neighbors (KNN)

The K-Nearest Neighbors (KNN) algorithm is a foundational, non-parametric, and instance-based supervised learning method widely used for both classification and regression tasks. Unlike "eager" learning algorithms that construct a generalized model from the training data before making predictions, KNN is a "lazy" learner. This means it defers the generalization process until a prediction query is made.

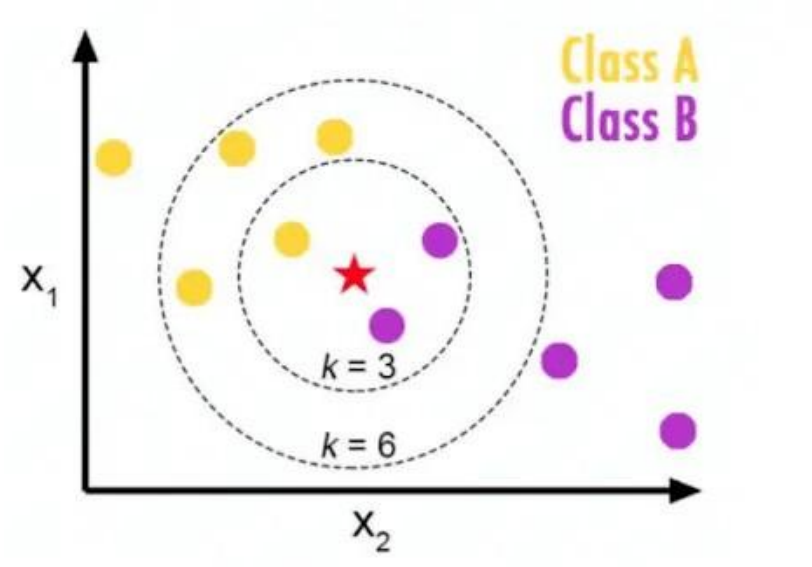


Figure 3.1 Illustration of the classification of K-Nearest Neighbors (KNN)

Figure 3.1 illustrates the classification of KNN. During its "training" phase, KNN simply stores all available cases and classifies new cases based on a similarity measure (distance function) with the stored instances.

3.1.1 KNN Architectures

KNN's architecture is a conceptual framework that centers on data instances and their measured proximity. Fundamentally, it's an instance-based learning approach where the entire training dataset serves as its architecture; KNN is a "lazy" learner that simply memorizes all instances without building an explicit model during training. For prediction, KNN operates by calculating the distance between a new query point and every training data point, using this metric to define "similarity" in the feature space. The K parameter, a hyper parameter, dictates the number of nearest neighbors to consider, with its optimal value typically found through cross-validation. Finally, the decision rule for classification assigns the new data point to the class determined by a majority vote among its K nearest neighbors, while for regression, it assigns the average (or weighted average) of the target values of those neighbors.

3.1.2 Some Challenges of KNN

Despite its simplicity, KNN faces several significant challenges. It incurs high computational costs during prediction, as it must calculate distances to all training points for every new query, making it slow for large datasets. Its performance degrades significantly with increasing features due to the curse of dimensionality, where distances become less meaningful. KNN is also highly sensitive to feature scaling, as features with larger ranges can disproportionately influence distance calculations, necessitating pre-processing. Furthermore, it is susceptible to outliers and noisy data, which can drastically skew local neighborhood-based predictions. Finally, selecting the optimal 'K' value is non-trivial and dataset-dependent, requiring extensive cross-validation to balance between sensitivity to noise (small K) and over-smoothing (large K).

3.2 Introduction to Support Vector Regression (SVR)

Support Vector Regression (SVR) is a powerful and versatile supervised learning algorithm used for regression analysis. It is an extension of the Support Vector Machine (SVM) classification algorithm, adapted to predict continuous-valued outputs instead of discrete class labels. The fundamental principle of SVR is to find a function that approximates the mapping from input features to a continuous target variable with a certain degree of accuracy, while aiming to minimize the prediction error and control model complexity. Unlike traditional regression models that try to minimize the squared error, SVR aims to find a function that has at most a deviation of ϵ from the obtained targets for all training data, and at the same time is as flat as possible. This concept of an ϵ-insensitive loss function is central to SVR.

3.2.1 SVR Architecture

SVR's architecture fundamentally involves mapping input features into a high-dimensional space through a kernel function (such as Linear, Polynomial, Radial Basis Function (RBF), or Sigmoid), which enables the model to capture non-linear relationships. In this transformed space, SVR then seeks to find a linear regression function. A key component is the ϵ-insensitive loss function, which only penalizes errors that fall outside a defined ϵ-margin, thereby promoting sparsity in the solution. The regularization parameter (C) balances minimizing this training error against maximizing the function's flatness. The learned regression function is primarily defined by the support vectors, which are the critical training data points lying on or beyond the ϵ-insensitive tube.

3.2.2 Some Challenges of SVR

Despite its power, SVR presents several challenges, primarily its high sensitivity to hyperparameter tuning, including the kernel type, its parameters (like γ), C, and ϵ, which often necessitates extensive, computationally expensive experimentation. Choosing the right kernel is critical and complex, as no single kernel is universally optimal, and an inappropriate choice can lead to underfitting or overfitting. SVR also incurs high computational costs and memory usage for large datasets, often scaling poorly with the number of samples, making it prohibitive for massive scales. Furthermore, its interpretability is limited due to the transformation into high-dimensional feature spaces, making it difficult to understand individual feature influence.

While robust to some noise via the ϵ-insensitive loss, it can still be affected by significant noisy data or outliers, impacting support vector determination, and its scalability to extremely high dimensions can also be challenging.

3.3 Introduction to Decision Tree (DT)

Decision Trees (DTs) are supervised learning algorithms used for both classification and regression tasks. They function by learning simple decision rules inferred from the data features. A decision tree visually resembles a flowchart-like structure where each internal node represents a test on an attribute (feature), each branch represents the outcome of the test, and each leaf node (terminal node) represents a class label (for classification) or a predicted value (for regression).

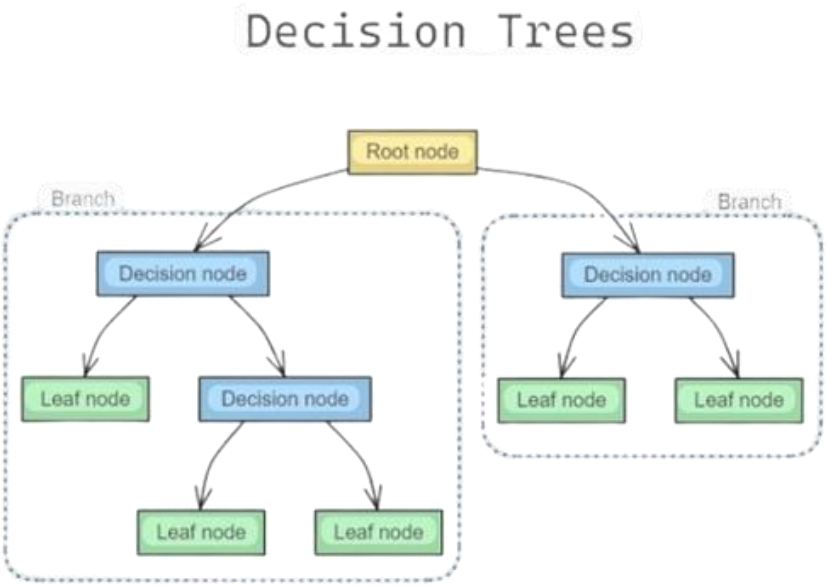


Figure 3.2 : Illustration of a DT (Classification)

Figure 3.2 illustrates the classification of DT. The path from the root to a leaf represents a classification or prediction rule. Decision trees are known for their interpretability and ability to handle both categorical and numerical data.

3.3.1 Decision Tree Architecture

Decision tree is hierarchical, beginning with a Root Node representing the entire dataset. From this root, and subsequently from Internal Nodes (also called decision nodes), tests on specific data attributes guide the data's path. Each test outcome forms a Branch, leading to further internal nodes or ultimately to Leaf Nodes (terminal nodes).

These leaf nodes represent the final prediction: a class label for classification trees (determined by majority rule) or a numerical value (typically the average of training instances) for regression trees. The process of splitting nodes is governed by Splitting Criteria, which aim to create child nodes with increased "purity" concerning the target variable. For classification, common criteria include Gini Impurity and Information Gain (based on Entropy), while for regression, Mean Squared Error (MSE) Reduction and Mean Absolute Error (MAE) Reduction are frequently used.

3.3.2 Some Challenges of Decision Trees

Decision Trees face several limitations. They are highly prone to overfitting, learning noise in deep structures which leads to poor generalization; this requires mitigation through pruning or depth constraints. They exhibit high variance, meaning small data changes can result in significantly different and unstable tree structures, a problem often addressed by ensemble methods. Decision trees can also struggle to capture very complex or smooth relationships efficiently, often forming step-wise decision boundaries. Their greedy construction process, which makes locally optimal splits without global foresight, may not yield the most optimal tree. Furthermore, data fragmentation can occur as trees grow deep, leading to unreliable statistics in small leaf nodes, and for problems with clear linear relationships, simpler models might perform better.

3.4 Introduction to XGBoost (XG-Boost)

XGBOOST (eXtreme Gradient Boosting) is a powerful and widely used gradient boosting algorithm that has achieved state-of-the-art results on many machine learning challenges, particularly for structured and tabular data. Developed by Tianqi Chen, XGBOOST is designed with both efficiency and flexibility in mind. It implements the gradient boosting framework but incorporates several optimizations and enhancements that contribute to its speed, scalability, and performance. It can be applied to both classification and regression problems and is known for its regularized model formalization, which helps prevent overfitting.

3.4.1 XGBoost Architecture

XGBoost's architecture is built upon the gradient boosting framework, sequentially constructing an ensemble of decision trees where each new tree corrects the errors of preceding ones, summing individual predictions for the final output. It minimizes a regularized objective function comprising a loss term (measuring prediction error) and a regularization term (penalizing tree complexity with L1 and L2 norms) to prevent overfitting. The tree learning algorithm greedily selects splits that maximize gain, defined by loss reduction and regularization. XGBoost employs efficient split-finding algorithms like the exact greedy approach or faster approximate methods (e.g., Weighted Quantile Sketch for candidate split points). Key system optimizations include parallel tree building (within each tree), cache-aware access, and out-of-core computing for scalability. Furthermore, it intelligently handles missing values by learning the optimal split direction and is designed with sparsity awareness for efficient processing of sparse data.

3.4.2 Some Challenges of XGBOOST

Despite its numerous strengths, XGBoost presents several challenges. Its performance is highly sensitive to hyperparameter tuning, which involves extensive and computationally expensive experimentation to find optimal combinations, and without careful tuning or early stopping, there's a risk of overfitting. The ensemble of many boosted trees makes XGBoost a complex "black box" model, limiting interpretability of specific prediction reasoning, and this black-box nature can hinder optimal use by non-experts. Furthermore, like all machine learning algorithms, its accuracy is sensitive to data quality, and training very large models still demands significant computational resources (CPU time and memory), even with its optimizations.

3.5 Introduction to Artificial Neural Network (ANN)

Artificial Neural Networks (ANNs) are a class of machine learning models inspired by the structure and function of the human brain. They are particularly effective in modeling complex, non-linear relationships between input and output variables, making them suitable for tasks such as classification, regression, and time series forecasting. An ANN consists of multiple layers of interconnected processing units called neurons, organized into an input layer, one or more hidden layers, and an output layer.

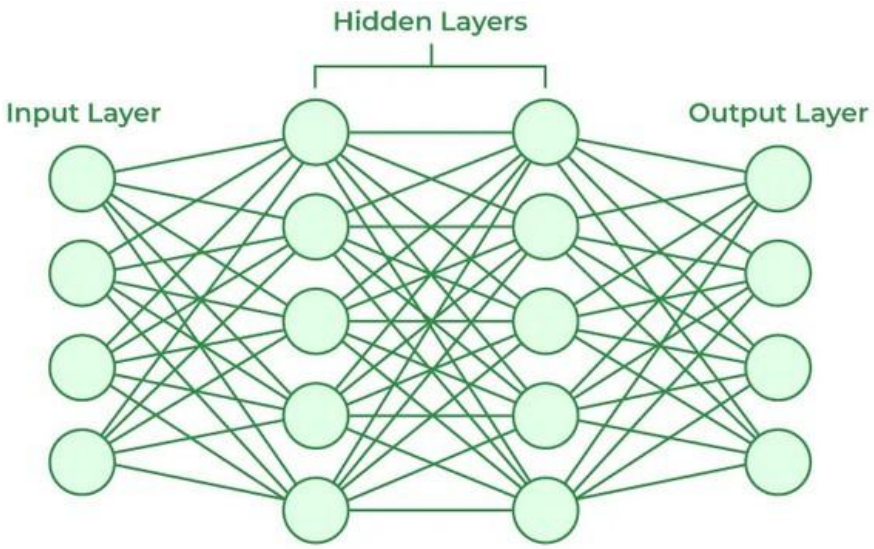


Figure 3.3: Illustration of ANN Layers

Figure 3.3 illustrates the ANN layers.Each neuron in the network receives a set of inputs, applies a linear transformation through weighted connections, adds a bias term, and then passes the result through a non-linear activation function.

3.5.1 Architecture of ANN

The architecture of an Artificial Neural Network (ANN) includes several key parts that work together to process data and make predictions. It starts with the input layer, which receives the raw data—each input node stands for one feature of the dataset. Next are the hidden layers, where the actual learning happens. These layers contain neurons that calculate weighted sums and apply activation functions like ReLU or sigmoid to capture patterns and relationships in the data. The output layer provides the final prediction, with the number of neurons matching the number of output classes or values. The connections between neurons have weights that determine how strongly signals are passed, and biases that help adjust the outputs. Activation functions add non-linearity, enabling the network to learn complex patterns. During training, data flows forward through the network (forward propagation), and the network compares its predictions to actual results using a loss function. Then, using backpropagation, it updates the weights and biases to improve accuracy by minimizing the loss.

3.5.2 Some Challenges of ANN

Artificial Neural Networks (ANNs) face several challenges. One major issue is overfitting, where the model learns the training data too well, including noise, and fails to generalize to new data—this requires techniques like dropout or regularization. ANNs also have a high computational cost, needing powerful hardware and long training times. They are sensitive to hyperparameters such as learning rate, number of layers, and activation functions, which require careful tuning. Another limitation is their black-box nature, making it difficult to interpret how decisions are made. Additionally, ANNs generally need large datasets to perform well, and in deep networks, they can suffer from vanishing or exploding gradients, which hinder training. Lastly, while ANNs can learn features, effective feature engineering still plays an important role in achieving good results.

3.6 Introduction to Random Forest (RF)

Random Forest (RF) is a powerful ensemble learning method applicable to both classification and regression. RF builds multiple decision trees during training and combines their outputs, either by majority vote for classification or by averaging for regression. This approach significantly boosts the stability, accuracy, and generalization of individual decision trees. RF specifically tackles the overfitting common in single decision trees through two core mechanisms: Bootstrap Aggregation (Bagging), which involves random sampling of data with replacement, and Feature Randomness, where only random subsets of features are considered when making split decisions within each tree.

3.6.1 Architecture of Random Forest (RF)

Random Forest (RF) model operates in distinct stages. First, Bootstrapping generates multiple diverse training datasets by randomly sampling with replacement from the original data, where each sample may contain duplicates and omit certain original data points. Next, for each of these bootstrap samples, a Decision Tree is constructed.

A crucial element here is that during each node split within a tree, only a random subset of features is considered, from which the optimal feature is selected for the split. Finally, for Aggregation of Results, the predictions from all individual trees are combined: for classification, the final prediction is determined by a majority vote among the trees, while for regression, it is the mean of all tree outputs

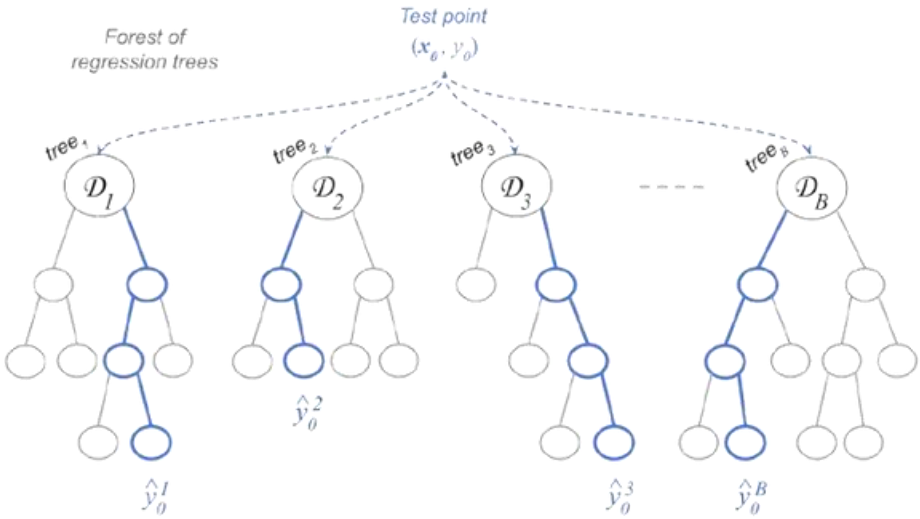


Figure 3.4: Illustration of ANN Layers

Figure 3.4 shows the ANN layers where each sample may contain duplicates and omit certain original data points. Next, for each of these bootstrap samples, a Decision Tree is constructed.

3.6.2 Some Challenges of of Random Forest (RF)

Though it has some advantages, Random Forest presents several challenges. It can have high computational complexity, requiring significant memory and time due to building multiple deep trees, especially with large datasets.

Although individual trees are interpretable, RF's ensemble nature reduces its overall interpretability, making the collective decision-making process less transparent.While generally robust, it can still overfit in the presence of significant noise or with very deep trees and small datasets. The model size can also become very large due to the numerous trees, which may limit its suitability for real- time or resource-constrained applications. Additionally, RF can be biased towards majority classes in imbalanced datasets without proper handling, and its feature importance metrics may exhibit bias towards features with higher cardinality or more levels.

3.7 Introduction to Approached Meta Model

Meta-modelling, also known as stacking or stacked generalisation, is a sophisticated ensemble learning method that combines the advantages of several different underlying machine learning models to increase prediction accuracy. Instead of combining predictions from homogenous models as in more straightforward ensemble techniques like Bagging (e.g., Random Forest) or Boosting (e.g., XGBOOST), stacking entails training a "meta- learner" (or blender) on the predictions produced by many “base learners”. The fundamental tenet is that certain base models perform better at handling particular kinds of mistakes or capturing various facets of the underlying data patterns. A meta model can frequently outperform any single base model in terms of generalization performance and resilience by figuring out how to best integrate these disparate predictions. In complicated fields like solar power forecasting, where the input-output relationships are extremely non-linear and impacted by a wide range of interacting factors, this method works especially well.

To increase prediction accuracy, the meta-model in this study employs a two- level stacking ensemble technique. Three foundational models are employed at the first stage: XGBoost, Random Forest, and Support Vector Regression (SVR). These models—SVR for managing non-linear relationships, RF for stability and resistance to overfitting, and XGBoost for fast speed, accuracy, and regularization—were selected due to their capacity to identify various patterns in the data. A Linear Regression (LR) model serves as the meta- learner at the second level.The entire process is wrapped in a Pipeline that first applies the defined preprocessor (handling numerical imputation and scaling, and categorical imputation and one-hot encoding) to the input data before feeding it to the stacking regressor. Cross-validation (with 5 folds) is used during the training of the stacking regressor to improve its robustness.

3.7.1 Architecture of Meta Model

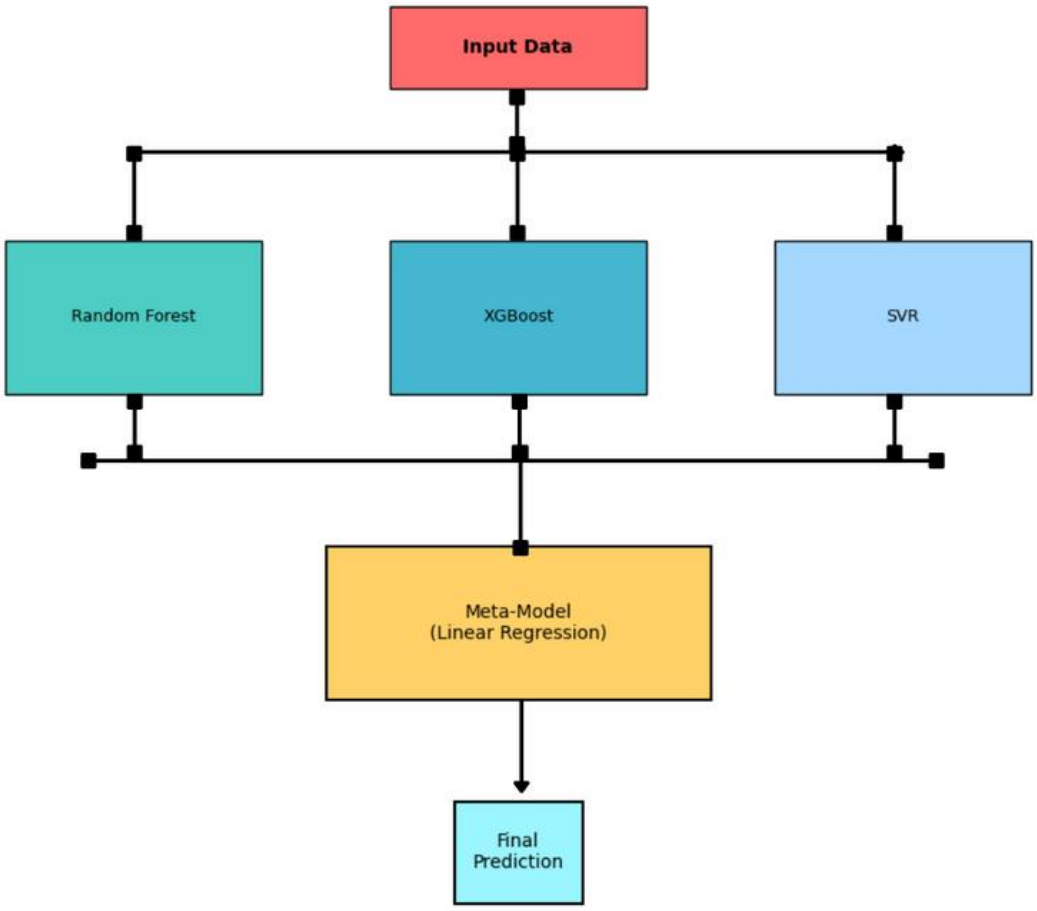


Figure 3.5 : Illustration of Approached Meta Model Layers

Figure 3.5 shows the structure of the proposed meta model. Instead of combining predictions from homogenous models as in more straightforward ensemble techniques like Bagging (e.g., Random Forest) or Boosting (e.g., XGBOOST), stacking entails training a "meta- learner" (or blender) on the predictions produced by many “base learners”.

To increase prediction accuracy, the meta-model in this study employs a two- level stacking ensemble technique. Three foundational models are employed at the first stage: XGBoost, Random Forest, and Support Vector Regression (SVR). These models—SVR for managing non-linear relationships, RF for stability and resistance to overfitting, and XGBoost for fast speed, accuracy, and regularization—were selected due to their capacity to identify various patterns in the data. A Linear Regression (LR) model serves as the

meta- learner at the second level. By allocating appropriate weights, LR effectively combines the outputs of the basis models, reducing overfitting and enhancing interpretability yet being simpler than the foundation models. The entire process is wrapped in a Pipeline that first applies the defined preprocessor (handling numerical imputation and scaling, and categorical imputation and one-hot encoding) to the input data before feeding it to the stacking regressor. Cross-validation (with 5 folds) is used during the training of the stacking regressor to improve its robustness.

3.7.2 Challenges of Meta Model

Using this type of stacked model comes with a few difficulties. First, it's more complex and takes longer to train because we're training several base models and then another model on top of them. This also makes tuning difficult, as we need to find the best settings for many different parts of the system all at once. There's also a risk of overfitting, meaning the model might become too good at predicting training errors and perform poorly on new data if not handled carefully. This approach also generally needs a lot of diverse data to train effectively. Furthermore, understanding why the model makes certain predictions can be hard, making it a bit of a "black box." Finding and fixing errors also becomes more complicated due to its layered structure. Lastly, sometimes the improvement gained might not be worth the extra effort if one of the simpler base models already works very well on its own.

Chapter 4: Methodology

The study begins with collecting meteorological data relevant to solar power generation, followed by preprocessing and normalization. Conventional machine learning models—including XGBoost, K-Nearest Neighbors (KNN), Support Vector Regression (SVR), Random Forest (RF), and Artificial Neural Networks (ANN)—are individually trained to predict solar power output. Their performance is evaluated using MAE, RMSE, MAPE, and R^2 metrics. To improve accuracy, a meta-model is developed by stacking the predictions of these base models into an ensemble. Finally, a comparative analysis is conducted to evaluate the performance differences between the meta-model and the individual conventional models.

4.1 Flow Chart for the Proposed Work

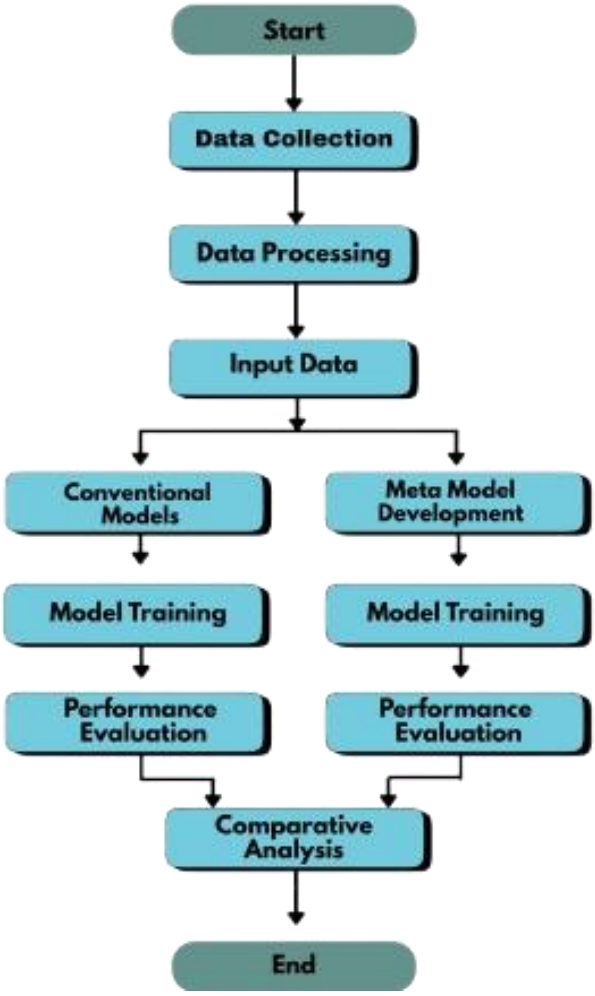


Figure 4.1: Flowchart of the proposed work

1. Start: The beginning of the model development process.

2. Data Collection : This step involves acquiring the historical dataset that will be used for training and evaluating the conventional and the meta-model. The data should contain relevant features and the target variable to be predicted.

3. Data Pre-processing : This crucial step prepares the dataset for modelling. It addresses issues like missing values, inconsistent formats, and feature scaling, encoding, ensuring the data is clean and suitable for machine learning algorithms.

4. Data Splitting (Train/Test) : The pre-processed dataset is divided into two subsets: a training set and a testing set. The training set is used to train models and the meta-learner, while the testing set is used to evaluate the final model's generalization performance. A typical split of 80% for training and 20% for testing is used.

5. Meta-Model Development : This section outlines the development of a stacked ensemble meta-model that leverages the strengths of Support Vector Regression (SVR), Random Forest (RF), and XG-Boost as base learners, with Linear Regression (LR) acting as the final predictor (meta-learner).

6. Model Training : This represents the simultaneous training of the individual conventional models and our approached meta model (SVR, Random Forest, XG-Boost). Each model learns to predict the target variable independently using the training data. End: The conclusion of the model development and deployment process.

7. Performance Evaluation : Machine learning model performance is evaluated using metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and R-squared (R^2). MAE and RMSE measure the average and squared differences between predicted and actual values, while MAPE expresses the error as a percentage, making it easier to interpret. R^2

indicates how well the model explains the variance in the data. Lower MAE, RMSE, and MAPE values, along with higher R^2, reflect better model accuracy and reliability in prediction tasks.

8. Comparative Analysis : The performance comparison between the individual models and the meta-model was conducted using standard evaluation metrics. While the individual models performed well in specific areas, their accuracy varied across different metrics. In contrast, the meta-model showed consistently better results, achieving improved scores across all evaluation metrics. This demonstrates that the meta-model offers more accurate and reliable predictions by effectively leveraging the strengths of the base models.

8. End : The conclusion of the model development and deployment process.

4.2 Data Collection

We have collected the historical datasets from Visual Crossing. It contains the daily average data of Sylhet, Bangladesh, from 2010 to 2024 of parameters like temperature, humidity, wind direction, solar radiation, etc. As machine learning algorithms have a hard time working with missing data values, the pre-processing stage began with addressing missing data to ensure the dataset's integrity. Data features missing values constituting less than 1% of the total data, the mean imputation was used to fill data values on the contrast. For higher percentages of missing values, more robust techniques such as median imputation and interpolation were applied. Additionally, outliers were used to toss out unusual data points that could have negatively affected the prediction. Finally, the correlation matrix was put to use to identify and eliminate features that were highly correlated. These pre-processing techniques were critical for creating clean datasets and help give accurate predictions.

4.3 Data Processing

The data processing phase involved several key steps to ensure the quality and usability of the dataset. Initially, raw meteorological data was collected and examined for missing or inconsistent values, which were then handled using appropriate imputation techniques. The data was cleaned and normalized to bring all features to a common scale, improving model performance and convergence. Irrelevant or redundant features were removed through feature selection methods to reduce noise and enhance model efficiency. Finally, the processed data was split into training and testing sets to facilitate model training and performance evaluation.

4.3.1 Scatterplot Analysis

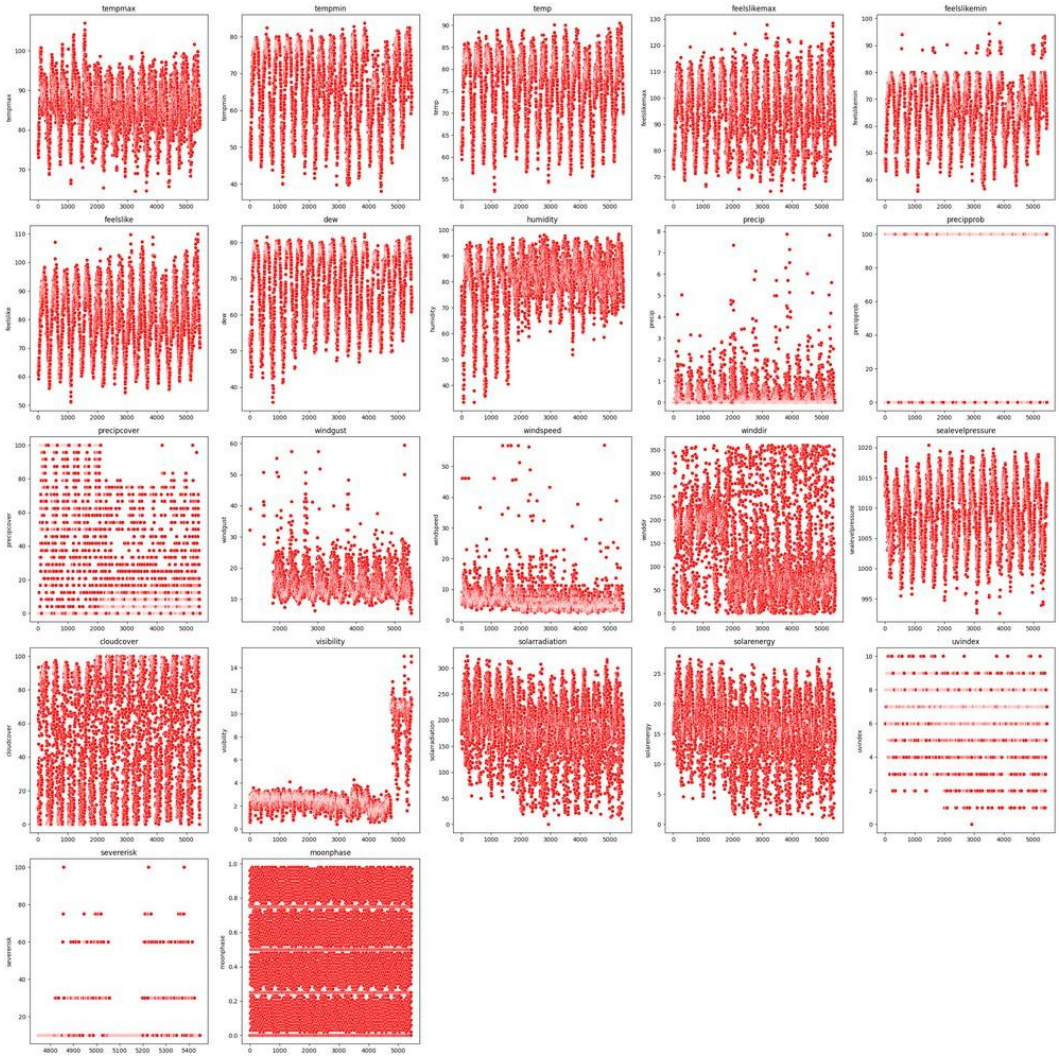


Figure 4. 2: Scatterplot of features

Figure 4.2 is a graphical representation used to visualize the relationship between two continuous variables. Each point on the plot represents an observation in the dataset, with its position determined by the values of the two variables on the X and Y axes. Scatter plots are particularly useful for identifying patterns, trends, clusters, and potential outliers, as well as for assessing whether a linear or nonlinear relationship exists between the variables. In predictive modeling and regression analysis, scatter plots help evaluate the correlation and distribution between features and the target variable.

- Temparature features (temp, tempmax, feelslikemax, feelslike, feelslikeavg) show a positive trend with solar energy, indicating that higher temperatures are often associated with higher energy values.
- Humidity exhibits a moderate inverse pattern, where higher humidity values generally correspond to lower solar energy, likely due to cloud coverage or moisture in the air.
- Wind related features (windgust, windspeed, winddir) and precipitation variables (precip, precipprob, precipcover) show scattered or weak patterns, suggesting limited direct correlation with solar energy.
- Cloudcover and visibility present distinct visual patterns: high cloud cover is associated with lower solar energy, while higher visibility tends to align with higher energy levels.
- Some categorical or constant-like features such as uvindex, severerisk, and moonphase display horizontal bands or repeated patterns, indicating limited variability or discrete value ranges which may require careful encoding or transformation.
- The scatter plot for solarradiation reveals a strong positive correlation, confirming its direct relationship with the target variable.

4.3.2 Correlation Analysis

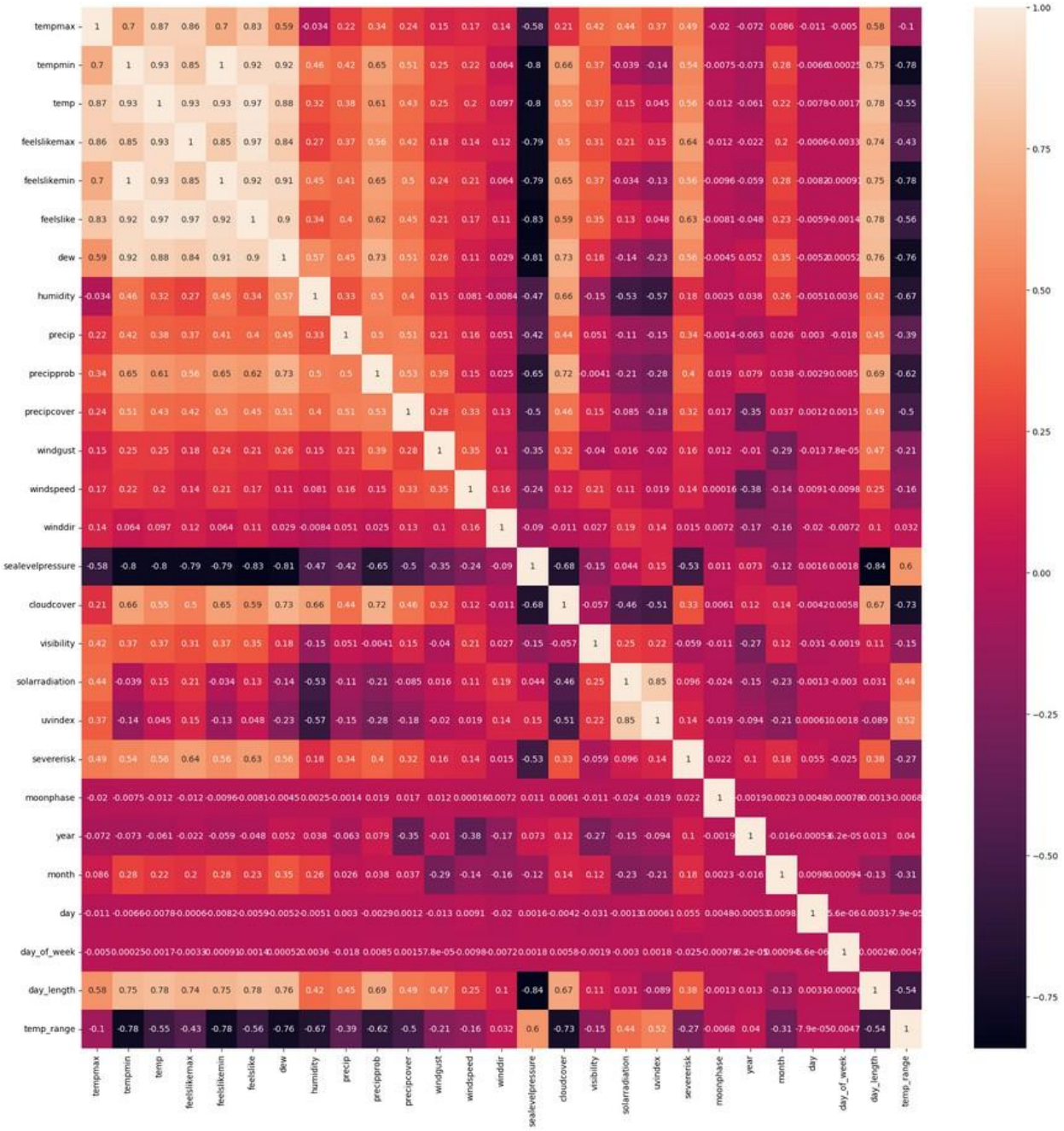


Figure 4.3: Correlation of features

Figure 4.3 shows the correlation between the features used in this study.The correlation heatmap provides a comprehensive overview of the relationships among the numerical variables in the dataset, with a particular focus on the target variable, solarenergy. The correlation coefficients range from -1 to 1, where values close to 1 indicate strong positive linear relationships, values close to -1 indicate strong negative relationships, and values near 0 imply little to no linear association.

Several features in the dataset demonstrate strong positive inter-correlations. Variables such as tempmax, tempmin, temp, feelslikemax, feelslikemin, feelslike, and dew are highly correlated with one another, with correlation coefficients often exceeding 0.85. This is expected, as these features represent various forms of temperature measurements and tend to move together.

In contrast, temp_range—which measures the difference between maximum and minimum temperature—shows a negative correlation with these variables, as it is derived from their variation.In relation to the target variable solarenergy, the heatmap reveals that uvindex has the highest positive correlation (approximately 0.85), indicating that greater ultraviolet intensity corresponds to higher solar energy values. Additionally, features such as tempmax, temp, feelslike, and day_length also exhibit moderate positive correlations with solarenergy, generally ranging between 0.4 and 0.5. This suggests that warmer temperatures and longer daylight durations contribute to increased solar energy accumulation.
Conversely, variables such as cloudcover, humidity, dew, sealevelpressure, and tempmin display moderate to strong negative correlations with solarenergy, typically in the range of -0.4 to -0.61. The negative correlation with cloudcover is particularly notable, as cloud presence directly reduces solar radiation reaching the surface. Similarly, higher humidity and dew point values may be associated with increased atmospheric moisture, which can reduce solar energy due to greater absorption and scattering.

Some features, including moonphase, year, month, day, and winddir, exhibit very low correlation with solarenergy and most other variables. These features may not significantly impact the prediction of solar energy and could potentially be excluded during feature selection. Overall, the correlation heatmap aids in identifying the most relevant meteorological variables that influence solar energy and supports the development of more accurate predictive models.

4.3.3 Handling Missing Values

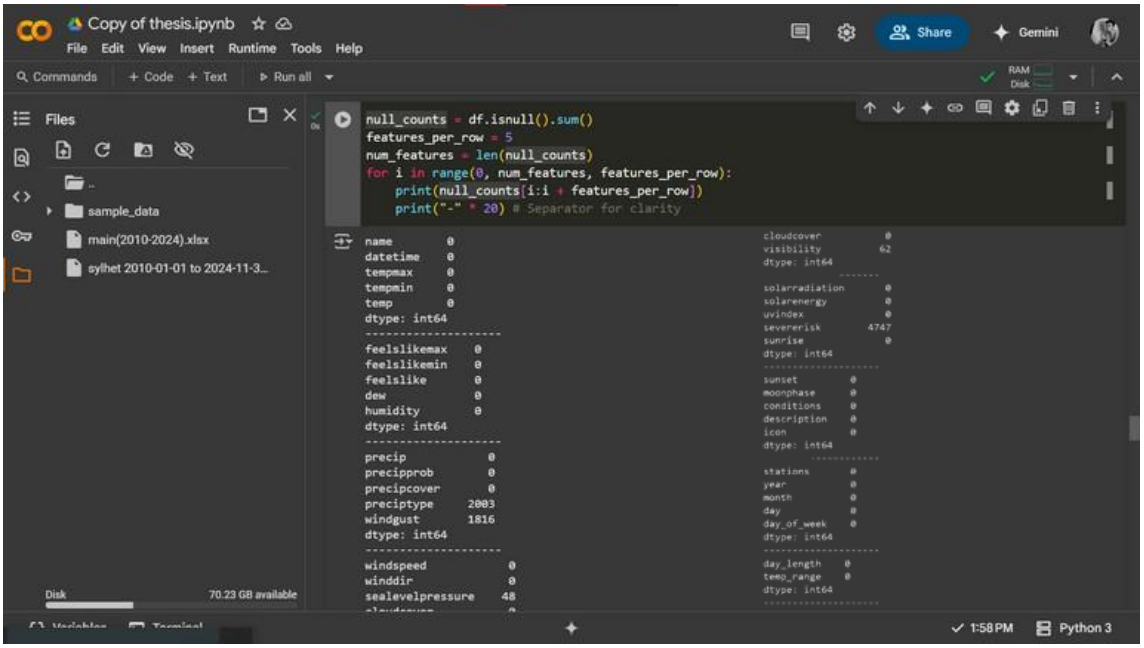


Figure 4.4: Imputation of Missing Values

Figure 4.4 shows the dealing of missing values of the used dataset. Common techniques include dropping rows or columns with excessive missing values, especially if they contribute little to the prediction task. For features with fewer missing entries, imputation methods such as replacing with the mean, median, or mode are often used.

In the given dataset, most features have no missing values, except for visibility with 62 missing entries and severerisk with 4747 missing entries, preciptype with 2003 missing entries, windgust with 1816 missing entries and sea-level pressure with 48 missing entries. Handling missing values is a crucial step in data preprocessing to ensure the accuracy and reliability of machine learning models. Common techniques include dropping rows or columns with excessive missing values, especially if they contribute little to the prediction task. For features with fewer missing entries, imputation methods such as replacing with the mean, median, or mode are often used. In time-series or sequential data, forward or backward fill can be applied. Alternatively, missing values can be replaced with a constant or predicted using other features in the dataset. In some cases, creating a new binary column to flag missing values helps capture any potential pattern in the missingness itself. The choice of method should depend on the importance of the feature and the overall impact on the dataset.

Figure4.5 : Handling Missing Values

Figure 4.5 shows the process of pre-processing the data in terms of handling the missing values with proper transformation step.

To ensure data quality and maintain consistency in preprocessing, missing values were addressed using a pipeline system. This approach enables a streamlined and automated sequence of data transformation steps, enhancing reproducibility and efficiency in model training.

For numerical features, missing values were handled using the SimpleImputer method with the "median" strategy. The median, being the middle value in a sorted list of numbers, is robust against outliers and provides a reliable central tendency measure for imputation. This helps to minimize the distortion that extreme values might cause if a mean-based strategy were used.

For categorical features, the SimpleImputer was applied with the "constant" strategy, where all missing entries were replaced with a predefined fixed value. This approach ensures uniform treatment of missing categorical data, commonly substituting missing values with labels such as 'unknown' or 'missing'. By integrating these imputation strategies within a pipeline framework, the preprocessing process remains consistent and efficient across different data subsets, ensuring the integrity of the dataset prior to model development.

4.3.4 Handling Outliers

Outliers are data points that deviate significantly from the general pattern of the data. In many real-world datasets, particularly in environmental and meteorological data, such as weather or solar energy datasets, outliers often arise due to sensor errors, data recording faults, or rare and extreme weather conditions. These extreme values can skew statistical summaries, mislead machine learning algorithms, and reduce the overall accuracy and robustness of predictive models. Therefore, it is essential to identify and manage outliers effectively before proceeding with further data analysis or model training.

Overview of the Dataset

The dataset used in this study contains both numerical and categorical features related to weather and solar energy parameters.The original features include:
'tempmax', 'tempmin', 'temp', 'feelslikemax', 'feelslikemin', 'feelslike', 'dew', 'humidity', 'precip', 'precipprob', 'precipcover', 'windgust', 'windspeed', 'winddir', 'sealevelpressure', 'cloudcover', 'visibility', 'solarradiation', 'solarenergy', 'uvindex', 'severerisk', 'moonphase', .

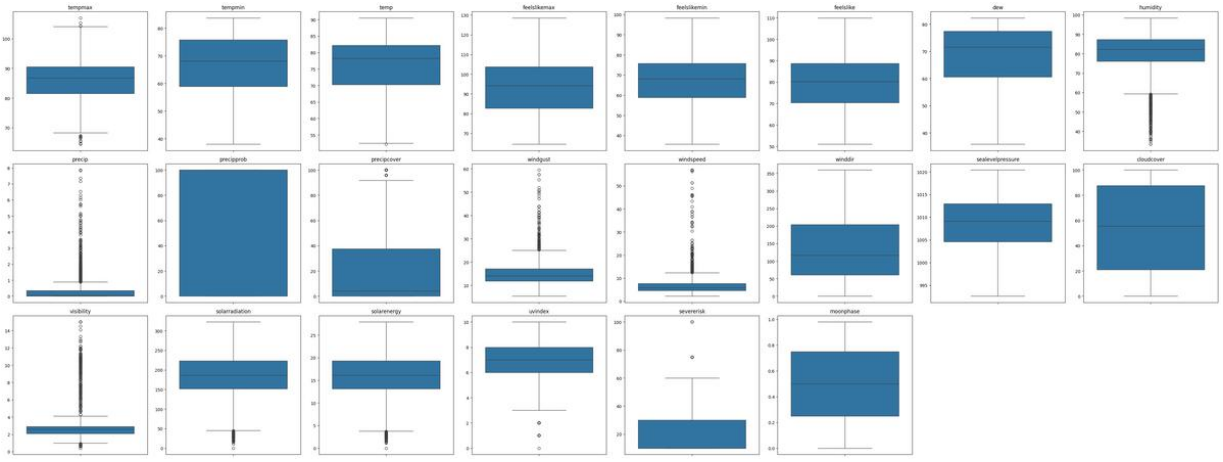


Figure 4.6: Boxplot of Features with Outlier

Figure 4.6 illustrates the features with outlier.Only numerical features were considered for outlier detection using the IQR method, as the technique is specifically designed for continuous numerical data. The categorical and timestamp-related fields were excluded since they are not compatible with statistical range-based detection techniques.

Selected Numerical Features for Outlier Treatment:

The IQR method was applied to the following numerical attributes:
Temperature-related: 'tempmax', 'tempmin', 'temp', 'feelslikemax', 'feelslikemin', 'feelslike' Atmospheric variables: 'dew', 'humidity', 'sealevelpressure', 'cloudcover', 'visibility' Precipitation-related: 'precip', 'precipprob', 'precipcover',Wind and motion: 'windgust', 'windspeed', 'winddir'.
Solar measurements: 'solarradiation', 'solarenergy', 'uvindex' Risk-related: 'severerisk'.

These variables are critical to understanding solar power generation and environmental conditions and must reflect accurate distributions for effective analysis.

Methodology: Interquartile Range (IQR) Approach

The steps followed for each numerical column were:
Calculate the first quartile (Q1) – the 25th percentile of the data.
Calculate the third quartile (Q3) – the 75th percentile.
Compute the Interquartile Range (IQR) using the formula:

$$IQR = Q3 - Q1 \quad (1)$$

Determine the lower and upper bounds:

$$\text{Lower bound} = Q1 - 1.5 \times IQR \quad (2)$$

$$\text{Upper bound} = Q3 + 1.5 \times IQR \quad (3)$$

The IQR method is a widely used statistical technique to detect and filter out extreme values. It is based on the concept of dividing a dataset into quartiles and identifying values that lie far from the interquartile spread.

Identify and remove outliers:

Any data point falling below the lower bound or above the upper bound was flagged as an outlier and removed from the dataset.

This method assumes that most data points fall within 1.5 times the IQR from the quartiles and that any deviation beyond this range is potentially anomalous or non- representative of the general data behavior.

Impact of Outlier Removal :

The removal of outliers had a noticeable impact on the quality and distribution of the data:

- Enhanced reliability of summary statistics: The mean and standard deviation became morerepresentative of the core dataset.

- Improved model performance: Machine learning algorithms trained on the cleaned data showed better generalization and reduced overfitting.

- Greater consistency across features: Extreme and unrealistic values were eliminated, helping models capture true patterns in the data.

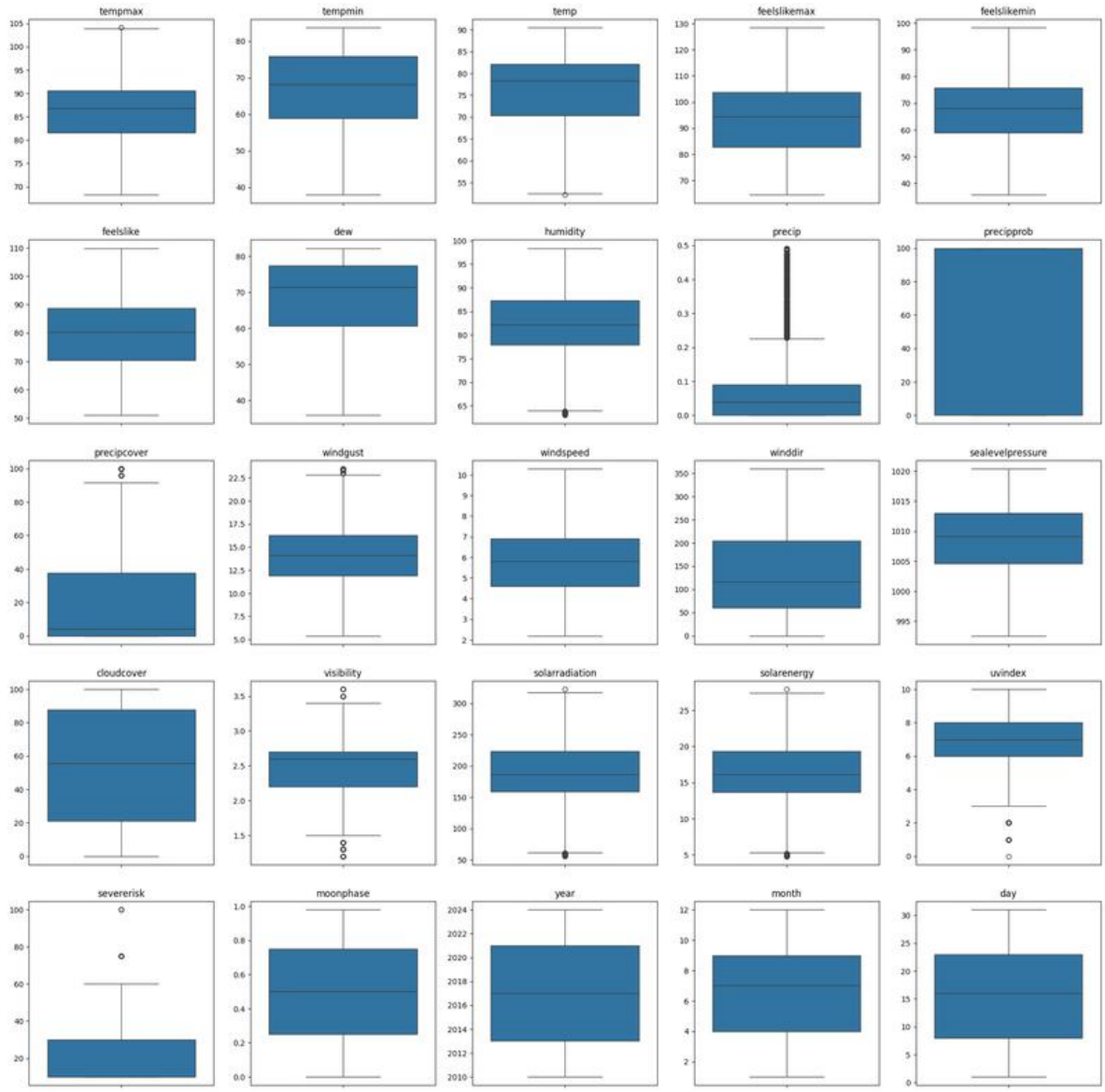


Figure 4.7 : Boxplot of features after removing the outlier

Figure 4.7 shows the boxplot of features after removing the outlier.This method assumes that most data points fall within 1.5 times the IQR from the quartiles and that any deviation beyond this range is potentially anomalous or non- representative of the general data behavior.

4.3.5 Feature Encoding

In this study, categorical features in the dataset were transformed using the One-Hot Encoding technique integrated into a pre-processing pipeline.. This method creates new binary columns for each category within a categorical variable, assigning a value of 1 if the observation belongs to that category and 0 otherwise. This approach prevents the introduction of any ordinal relationship among categories, which is crucial for maintaining the integrity of categorical data in machine learning algorithms that assume numerical inputs. By applying One-Hot Encoding, we ensured that the model could interpret categorical variables effectively without making any implicit assumptions about their relative importance or ranking.The encoding process was implemented through the sklearn.pipeline.Pipeline, which ensures a streamlined and reproducible workflow. Specifically, categorical columns were first selected and passed through the OneHotEncoder, which automatically handled unknown categories during transformation to maintain consistency across training and testing phases. This step was essential for enhancing the model's performance and reliability, particularly in scenarios involving algorithms sensitive to feature scaling and data representation, such as linear regression, support vector regression, and tree-based models.

```
# --- Preprocessing Pipelines ---
# Numeric pipeline: impute missing values with median and then standardize.
numeric_pipeline = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])

# Categorical pipeline: impute missing values and apply one-hot encoding.
categorical_pipeline = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])

# Combine both pipelines into a ColumnTransformer.
preprocessor = ColumnTransformer(transformers=[
    ('num', numeric_pipeline, numerical_features),
    ('cat', categorical_pipeline, categorical_features)
])
```

Figure 4.8 : OneHotEncoder used for categorization

4.4 Model Development

We developed a stacked ensemble learning model integrating diverse base learners—Support Vector Regression (SVR), Random Forest (RF), and Extreme Gradient Boosting (XG-Boost)—with a meta-learner implemented via Linear Regression (LR). This hybrid model aims to leverage the unique strengths of each algorithm to enhance predictive accuracy and generalization performance across complex, nonlinear datasets.

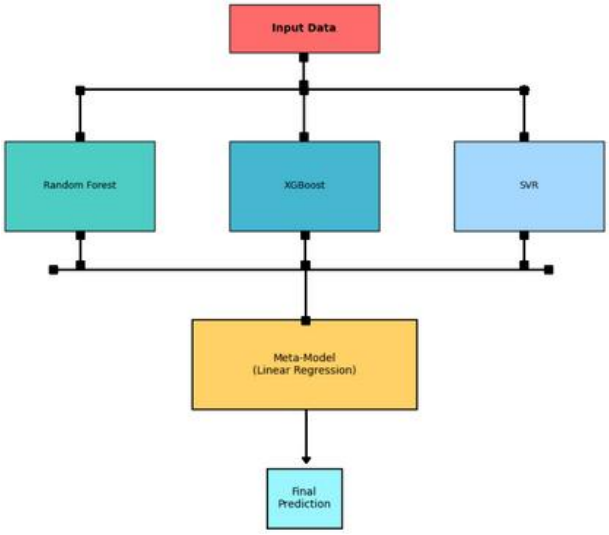


Figure 4.9: Meta-model Framework

I. Base Learners :

Three machine learning algorithms were selected as level-0 learners due to their complementary modeling characteristics:

Support Vector Regression (SVR):

Captures nonlinear patterns via kernel trick (RBF used). Robust to high-dimensional spaces and effective in small-to-medium datasets.

Random Forest (RF):

An ensemble of decision trees with bootstrapped aggregation and random feature selection. Handles nonlinearity well and reduces overfitting through averaging.

Extreme Gradient Boosting (XG-Boost):

Utilizes gradient-boosted decision trees with regularization. Known for high efficiency and performance in structured data prediction. Each base learner was trained on the same training set and tuned via grid search and cross-validation to optimize hyperparameters.

II. Meta-Learner (Stacking Strategy)

A Linear Regression model was employed as the level-1 meta-learner to combine the predictions from the base models:

Phase 1 – Base Model Training:

The training data were split using k-fold cross-validation (k=5). Each base model was trained on k-1 folds and predicted on the held-out fold.

This resulted in out-of-fold predictions for the entire training set from each base model.

Phase 2 – Meta-Model Training:

The out-of-fold predictions (three per training instance: from SVR, RF, and XG-Boost) were used as features to train the meta-model.

The original target variable remained the prediction target.

A simple Linear Regression was used to minimize variance and maintain interpretability.

Final Prediction:

For testing, base learners were retrained on the full training data and used to generate predictions for the test set. These predictions formed the input features for the trained meta-learner to produce the final predictions.

Model Evaluation Strategy

Model performance was assessed using stratified k-fold cross-validation to ensure robust estimation of generalization performance. The ensemble model's predictions were benchmarked against individual base models to quantify the benefit of meta-learning.

Chapter 5: Result and Comparative Analysis

This section presents the experimental results obtained from the evaluation of various machine learning models applied to solar energy prediction. The models were assessed using standard evaluation metrics including Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and the Coefficient of Determination (R^2). These metrics provide a comprehensive overview of each model's predictive accuracy and reliability. A comparative analysis has been conducted to identify the most effective model in terms of performance. The models tested include Random Forest (RF), XGBoost, Decision Tree (DT), Support Vector Regression (SVR), K-Nearest Neighbors (KNN), Artificial Neural Network (ANN), and a Meta model combining selected predictors. The purpose of this comparison is to highlight the strengths and weaknesses of each algorithm under the same dataset and evaluation criteria, thereby offering insight into their suitability for solar power forecasting tasks.

5.1 Actual vs Predicted Curve

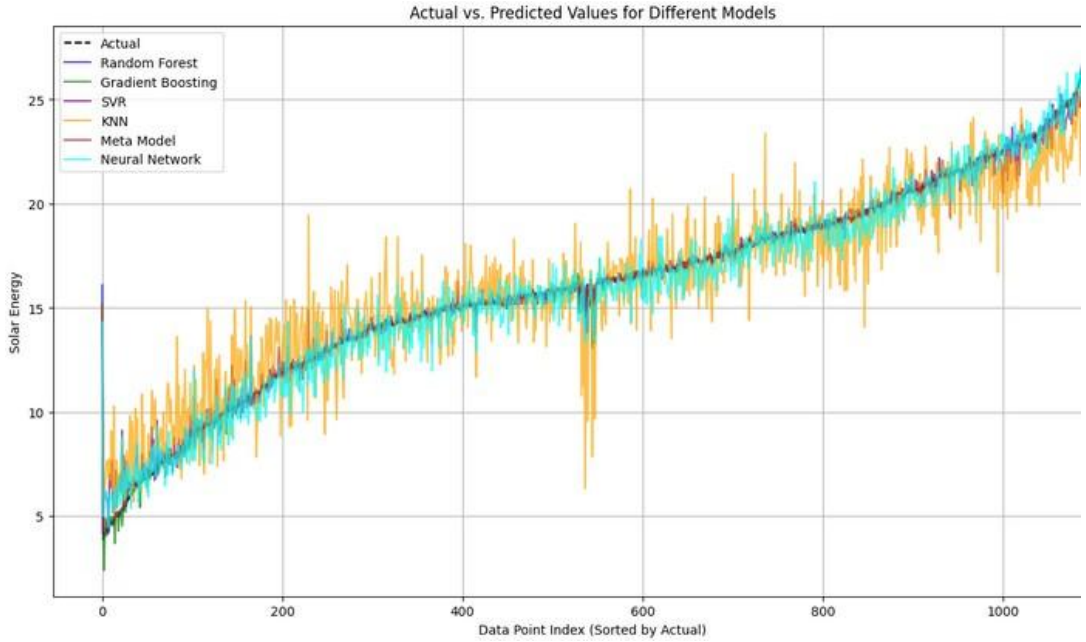


Figure 5.2: Convergence curve for daily load prediction model

In **Figure 5.2,** the visualization showcases red curves representing the predicted demand for a specific hour, contrasted with blue curves depicting the actual observed demand for that same hour. This visual comparison provides a means to gauge the predictive model's alignment with real-world demand patterns.

5.2 Evaluation Matrices

To evaluate the effectiveness of different machine learning models in predicting solar power generation, we conducted a performance comparison using several commonly used metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and the coefficient of determination (R^2). The models tested include Random Forest (RF), XG-Boost, Decision Tree (DT), Support Vector Regression (SVR), K-Nearest Neighbors (KNN), Artificial Neural Network (ANN), and a Meta-model built by combining multiple base learners. Each model was trained and tested using the same dataset for fair comparison. The evaluation focused on identifying which model could provide the most accurate and reliable solar power predictions.

The Mean Absolute Error (MAE) is defined as the average of the absolute differences between predicted values and actual values. It provides a straightforward measure of the magnitude of errors in a prediction without considering their direction. MAE is easy to interpret and is expressed in the same units as the predicted variable. A lower MAE indicates a more accurate model.

The Root Mean Square Error (RMSE) represents the square root of the average of squared differences between predicted and actual values. Unlike MAE, RMSE gives higher weight to larger errors due to the squaring term. This makes RMSE especially useful in scenarios where large errors are particularly undesirable. A lower RMSE value signifies a model that closely fits the data.

The Mean Absolute Percentage Error (MAPE) calculates the average absolute error as a percentage of the actual values. It normalizes error values, making it particularly useful when comparing model performance across datasets of different scales. A lower MAPE percentage indicates better model performance. However, MAPE can be sensitive when actual values are near zero, potentially inflating the error percentage.

The Coefficient of Determination (R^2) measures the proportion of the variance in the dependent variable that is predictable from the independent variables. It indicates how well the predicted values approximate the actual values. An R^2 value closer to 1 implies a stronger correlation and a better-performing model, whereas a value closer to 0 .

These metrics collectively offer a comprehensive understanding of a model's predictive capabilities, capturing both the magnitude and the consistency of its errors.

5.3 Evaluation Metrics Result

In terms of performance, the Random Forest and XG-Boost models both demonstrated high accuracy, with MAE values of 0.112 and 0.136, respectively, and RMSE values close to 0.419. Their R^2 scores were also very strong, both at 0.992, indicating a high level of fit between the predicted and actual values.The Decision Tree model followed closely, with a slightly higher RMSE of 0.523 and a slightly lower R^2 of 0.988.

ML Model	MAE	RMSE	MAPE	R^2
RF	0.112	0.419	0.016	0.992
XGBoost	0.136	0.416	0.013	0.992
DT	0.136	0.523	0.012	0.988
SVR	0.332	0.604	0.031	0.984
KNN	1.281	1.722	0.099	0.869
ANN	0.442	0.662	0.035	0.981
Meta	0.0974	0.3803	0.0090	0.9936

Figure 5.2: Results & Comparative Analysis

Figure 5.2 shows the result of the conventional models and the meta model.The Random Forest and XG-Boost models both demonstrated high accuracy, with MAE values of 0.112 and 0.136, respectively, and RMSE values close to 0.419. Their R^2 scores were also very strong, both at 0.992, indicating a high level of fit between the predicted and actual values.The Decision Tree model followed closely, with a slightly higher RMSE of 0.523 and a slightly lower R^2 of 0.988.

Support Vector Regression (SVR) showed moderate performance with increased error values (MAE: 0.332, RMSE: 0.604) and an R^2 of 0.984. In contrast, the KNN model had the weakest results with significantly higher MAE (1.281) and RMSE (1.722), and a low R^2 of 0.869. The Artificial Neural Network (ANN) also underperformed compared to tree-based models, recording an MAE of 0.442 and an RMSE of 0.662. Notably, the Meta-model outperformed all individual models, achieving the lowest MAE (0.0979), lowest RMSE (0.3803), lowest MAPE (0.0090), and the highest R^2 (0.9936), making it the most reliable and accurate model in this study.

5.4 Summary

In summary, the performance analysis demonstrates that the Meta-model delivers superior results across all evaluation metrics. Integrating the strengths of multiple base models (likely including RF, XG-Boost, and others) reduces error and enhances prediction stability. Random Forest and XG-Boost also showed excellent individual performance and are suitable for reliable solar forecasting. Models like KNN and ANN were less effective and showed higher error and lower consistency. Overall, the findings confirm that ensemble-based meta-modelling can significantly improve predictive accuracy in solar energy estimation tasks.

Chapter 6: Conclusion and Future Research Directions

6.1 Conclusion

Based on the evaluation of seven machine learning models for solar energy prediction, the comparative results demonstrate clear differences in performance across various metrics. Among the models tested, the Meta model exhibits the highest level of accuracy and reliability, outperforming all individual models in every evaluation criterion.

The Meta model achieved the lowest Mean Absolute Error (MAE) of 0.0979, indicating its predictions deviated the least from the actual values on average. It also recorded the lowest Root Mean Square Error (RMSE) of 0.3803, reflecting a reduced sensitivity to large errors compared to other models. Furthermore, the Meta model's Mean Absolute Percentage Error (MAPE) was just 0.0090, emphasizing its ability to maintain a very low relative prediction error. Most significantly, it achieved the highest Coefficient of Determination (R^2) of 0.9936, confirming that the model could explain over 99% of the variance in solar energy output.

Among the traditional models, Random Forest also performed strongly, with an MAE of 0.112, RMSE of 0.419, MAPE of 0.0106, and R^2 of 0.992, closely following the Meta model in overall performance. XG-Boost yielded nearly equivalent results to Random Forest with an R^2 of 0.992 and slightly higher errors. Decision Tree (DT) showed moderate accuracy with an R^2 of 0.988, but its higher RMSE and MAE values compared to RF and XG-Boost indicate less precise forecasting. Support Vector Regression (SVR) and K-Nearest Neighbors (KNN) displayed weaker performance, especially KNN, which had the highest MAE (1.281) and RMSE (1.722), and the lowest R^2 (0.869), suggesting its unsuitability for this task. The Artificial Neural Network (ANN) model performed moderately with an R^2 of 0.981 but was outperformed by tree-based models and the Meta approach.

Overall, the findings of this analysis highlight the superiority of ensemble and hybrid approaches in solar energy forecasting. The Meta model's enhanced ability to generalize and capture complex patterns by combining multiple models proves particularly beneficial in handling the non-linearities and uncertainties inherent in solar energy data. These results not only confirm the robustness of the proposed meta-modeling approach but also offer valuable insights for the design of future energy prediction systems that aim for higher precision and reliability in power system operations and planning.

6.2 Future Research Directions

In discussing future work, emphasis can be placed on the ongoing development and enhancement of the short-term load forecasting model for the Sylhet Division of Bangladesh. This involves several key aspects:

1. **Integration of Climate Uncertainty and Real-time Data**

Incorporate real-time satellite or IoT sensor data to update solar energy forecasts.

2. **Hybrid and Explainable Meta-Models**

Combine meta-learning with interpretable models (e.g., SHAP, LIME).Design hybrid meta-models that integrate deep learning with ensemble learning.

3. **Region-Specific and Seasonal Adaptability**

Develop adaptive models that recalibrate based on weather patterns and seasonal changes.
Explore transfer learning techniques to apply models from one region to another with limited labeled data.

4. **Integration with Satellite Imagery and Remote Sensing**

Use convolutional neural networks on satellite imagery for spatially informed solar potential prediction.
Combine image-based data with traditional weather or irradiation features in a multimodal meta-model.

5. **Application in Smart Grid and Energy**

Extend your prediction model to aid in dynamic energy storage management or smart grid load balancing.
Pair predictive outputs with optimization algorithms to allocate solar power in distributed energy systems.

6. **Model Robustness and Energy Efficiency**

Investigate lightweight models that require less computational power but maintain high performance—important for edge computing applications.

7. **Environmental and Socioeconomic Impact Forecasting**

Use solar predictions to model downstream effects, such as carbon offset, cost savings, or policy impacts.

Integrate GIS tools to highlight regions with high solar potential but low current adoption rates.

References

[1] A. K. Chaaban and N. Alfadl, 'A comparative study of machine learning approaches for an accurate predictive modeling of solar energy generation', Energy Reports, vol. 12, pp. 1293–1302, 2024

[2] Y. Ledmaoui, A. El Maghraoui, M. El Aroussi, R. Saadane, A. Chebak, and A. Chehri, 'Forecasting solar energy production: A comparative study of machine learning algorithms', Energy Reports, vol. 10, pp. 1004–1012, 2023.

[3] I. K. Tanoli et al., 'Machine learning for high-performance solar radiation prediction', Energy Reports, vol. 12, pp. 4794–4804, 2024.

[4] N. Saxena et al., 'Hybrid KNN-SVM machine learning approach for solar power forecasting', Environmental Challenges, vol. 14, p. 100838, 2024.

[5] E. Sarmas, E. Spiliotis, E. Stamatopoulos, V. Marinakis, and H. Doukas, 'Short-term photovoltaic power forecasting using meta-learning and numerical weather prediction independent Long Short-Term Memory models', Renewable Energy, vol. 216, p. 1189 97,2023.

[6] B. Zazoum, 'Solar photovoltaic power prediction using different machine learning methods', Energy Reports, vol. 8, pp. 19–25, 2022.

[7] K.Anuradha, Deekshitha Erlapally, G. Karuna, V. Srilakshmi and K. Adilakshmi, 'Analysis of Solar Power Generation Forecasting Using Machine Learning Techniques.' Energy Reports, vol. 10, pp. 1004–1012, 2023.

[8] Halima Haque, Md. Abdur Razzak. Medium-Term Energy Demand Analysis Using Machine Learning: A Case Study on a Sub-District Area of a Divisional City in Bangladesh. IEEE access 4424 – 4432.

[9] I. K. Tanoli et al., 'Machine learning for high-performance solar radiation prediction', Energy Reports, vol. 12, pp. 4794–4804, 2024.

[10] Y. Ledmaoui, A. El Maghraoui, M. El Aroussi, R. Saadane, A. Chebak, and A. Chehri, 'Forecasting solar energy production: A comparative study of machine learning algorithms', Energy Reports, vol. 10, pp. 1004–1012, 2023.