Solar Production and Load Prediction Competition Project Report

Muhammad Faiz Raza

Contents

1 Introduction

1.1 Project Overview

This work mainly focuses on developing precise predictive models for forecasting photovoltaic generation and load consumption. Accurate prediction of energy generation and consumption is vital in helping to optimally distribute energy as renewable energy sources, including solar power, gain significant importance within the energy grid, optimize efficiency and drive down operational costs. This, in turn, will improve accuracy in predictions and realize highly effective energy management and planning using the LightGBM and GRU models. The project involved broad feature engineering, hyperparameter tuning, and evaluation to ensure the reliability and robustness of the models.

1.2 Purpose and Scope

Solar energy, together with other renewable energies, is increasingly coming to play a pivotal role in global climate change mitigation and reducing reliance on fossil fuels. Integrating these renewable sources within power grids creates numerous challenges, primarily because energy generation is intermittent and variable. Accurate predictions of photovoltaic (PV) generation and load consumption are made for an efficient energy management procedure so that supply would meet demand and waste of energy is reduced. The project's objective is to develop a robust predictive model that can accurately forecast PV generation and load consumption. These predictions are really important for several reasons:

- **Resource Optimization**: Predicting energy generation and consumption in advance can allow utility companies to allocate resources, better maintenance plan, and fully use renewable energy sources.

- **Cost Efficiency**: Improved predictions enhance the energy trading strategies with cost savings in several aspects, such as operational expenditures, increasing economic viability for renewable energy, etc.

- **Sustainability**: Enhanced accuracy of energy prediction will contribute to sustainability. Larger goals it can contribute to are improved renewable integration and reduced greenhouse gas emissions.

Advanced machine-learning techniques are applied to the historical data for PV generation and load consumption, including developed time-based features, to improve the accuracy of predicting those target functions. The scope of the project includes data preprocessing, feature engineering, model development, and evaluation—all to be performed with the objective of obtaining a predictive model that can be used in practical energy management. Through increased accuracy in energy prediction, this project supports resource optimization towards a more sustainable and efficient energy system.

2 Challenges

Accurate prediction of PV generation and load consumption is very challenging due to several reasons:

- **Variability of Weather**: Solar power generation depends on various weather conditions, which are highly variable and difficult to predict with high-level accuracy. The main challenge in building predictive models comes from fluctuations in PV generation due to changes in sunlight intensity, cloud cover, and temperature.

- **System Inefficiencies**: Inefficiencies in PV systems, such as those caused by aging equipment or dust accumulation on panels, can negatively impact the accuracy of generation predictions. Often, these inefficiencies are not directly measurable, adding to the complexity of the modeling process.

- **Quality of Data**: The accuracy of predictions is highly sensitive to the data quality used for training models. Inconsistent data, missing values, and incorrect measurements can degrade model performance. This becomes especially challenging when integrating data from multiple sources like weather stations and energy meters.

- **Data Collection**: Accurate and comprehensive data collection on weather conditions is crucial for reliable predictions. This includes gathering data on solar irradiance, temperature, wind speed, and other meteorological factors influencing PV output.

- **Time to Train**: Training machine learning models, particularly when dealing with large datasets and complex models like GRUs or LightGBM, can be computationally expensive and time-consuming. Efficient management of computational resources and optimization of the training process are necessary to achieve timely results.

- **Overfitting**: With highly variable data, there is a risk of overfitting the model to the training data, particularly when using deep learning techniques. Overfitting occurs when the model learns noise or random fluctuations in the training data rather than the underlying patterns. Techniques like cross-validation, early stopping, and regularization are important to mitigate this risk.

2.1 Objective

The primary objective is to improve the accuracy of photovoltaic (PV) generation and load consumption forecasting through enhanced machine learning. An accurate prediction of energy generation and consumption becomes very pivotal about the optimal integration of renewable energy sources into the power grid, efficient distribution of energy, reduction of operational costs, and realization of grid stability.

In this regard, the project will develop models for robust predictive models that use state-of-the-art machine learning algorithms, such as gradient boosting machines (e.g., LightGBM) and recurrent neural networks such as GRU (Gated Recurrent Units), for capturing the intricate patterns and relationships between historic photovoltaic generation and load consumption.

- **Implement Comprehensive Feature Engineering**: Extract and engineer all the relevant features from raw data, including time-based features such as hour of the day, day of the week, seasonality, and parameters associated with weather, in a manner that provides richness in information to help predictive capacities for models.

- **Optimize Model Performance**: Rigorously use techniques for hyperparameter tuning and validation to optimize model configurations to minimize overfitting while maximizing generalization to new data scenarios.

- **Evaluate and Compare Model Efficacy**: Check the models' performance systematically with the required metrics, such as Mean Absolute Error and Root Mean Squared Error, to assert what approach would be the most effective way of accurate forecasting.

- **Real-World Applicability**: The developed models are supposed to be precise, efficient, and scalable for their deployment into real-time energy management systems and decision-making processes.

The project, through fulfilling these specific objectives, aims to contribute to more reliable and efficient management of renewable energy that supports more significant transitions to sustainable energy systems. More accurate prediction can help energy providers and grid operators optimize their forecast supply and demand fluctuations and resource allocation and, at the end of the day, provide support for power system stability and resilience.

3 Methodology

3.1 Data Collection

In this phase, historical data regarding photovoltaic generation and load consumption were sourced from various sources. This dataset contains timestamps, system identifiers, values of generation

(`generation_W`), and load values (`load_W`). The set of other variables that were gathered were weather data—temperature, wind speed, pressure—since these have a large influence on the generation and consumption of energy. The weather data was gathered from the Meteostate.

3.2 Preprocessing of Data

A significant amount of preprocessing was done before feeding the models.

- **Data Cleaning**: Observations with missing data within the dataset were either imputed or excluded based on the level of missingness and the possible impact on the model's accuracy. Outliers were identified and handled correctly so as not to disturb the performance of the model. In the train data, 79551 duplicates removed. 742985 indices were missing. Outliers removed. The Missing indices were added, to fill there values **interpolation** was used.

- **Feature Engineering**: New features have been created from the existing data of **timestamp** to enrich the dataset with additional context further and enhance the model's performance. New features $['year','month','day','hour','minutes','dayofweek','hour_sin','hour_cosine','month_sin','month_cosine'']$

3.3 Feature Engineering

Feature engineering was critical to boosting model prediction capability. The key features derived from the timestamp included Year, Month, Day, Hour, Minute, and Day of the Week. These time-based features capture seasonal, monthly, weekly, and daily patterns in PV generation and load consumption for appropriate prediction.

- **Weather Features**: The dataset included average temperature ($tavg$), minimum temperature ($tmin$), maximum temperature ($tmax$), precipitation ($prcp$), wind direction ($wdir$), wind speed ($wspd$), and atmospheric pressure ($pres$). They serve as a context for influencing energy generation directly.

3.4 Model Selection

The models selected are based on their proven record for time series forecasting and structured data:

- **LightGBM**: This efficient and scalable gradient boosting framework makes it pretty appropriate for big datasets. LightGBM works very well in capturing complex interactions of features, which is important in predicting PV generation and load. Native handling of categorical features, high training speed, and robust performance make it a very good candidate.

- **GRU**: Gated Recurrent Unit is a type of RNN that specializes in modeling sequential data and capturing temporal dependencies. Considering that the PV generation and load consumption present specific temporal patterns, the capability of GRU to store long-term dependencies without vanishing gradients made it suitable for this project.

3.5 Why LightGBM?

LightGBM was selected for several key reasons:

1. **Computational Efficiency**: - LightGBM is highly efficient and can handle large datasets with lower memory usage and faster processing times compared to many other models.

2. **Ease of Implementation**: - It offers a user-friendly implementation with effective default hyperparameters, making it easier to set up and use.

3. **Performance**: - LightGBM consistently delivers strong predictive performance. Its boosting mechanism iteratively improves predictions by learning from previous errors, reducing the Mean Absolute Error (MAE) effectively.

4. **Handling Imbalanced Data**: - The model handles imbalanced datasets well, which is beneficial for predicting PV generation and load with varying demands.

Comparison with GRU:

- **Computational Complexity**: GRU models are more computationally intensive and complex to implement, requiring more time and resources for training. - **Training Time**: GRU models often take longer to train due to their complex architectures, while LightGBM provides faster training and efficient performance.

3.6 Hyperparameter Tuning

To tune the hyperparameters, a systematic approach was applied to maximize the performance of selected models.

- **Grid Search**: A grid search explored combinations in the number of boosting rounds, learning rate, maximum depth, and the number of leaves in LightGBM, and the number of layers, hidden units, and learning rate for the GRU model. This left no stone unturned in the defined hyperparameter space and tuned the models to maximal predictive accuracy. But it's so much time consuming.

- **Manually Tuning** To mitigate the effect of time consumption grid-search. Manual hyper-tuning has been done here. By changing the hyper-parameters and observing the results.

This included rigorous preprocessing, careful model selection, and meticulous hyperparameter tuning. This formed a very solid base, on which one could then build precise and robust predictive models that can accurately foresee PV generation and load consumption.

4 Functional and Non-Functional Requirements

4.1 Functional Requirements

- **Prediction Capability**: The system should be able to predict PV generation and load consumption accurately based on the input data, including processing of time series.

- **Data Preprocessing**: The system should perform other data preprocessing tasks, including feature engineering, scaling, normalization, and lag feature creation. It must clean the data by handling missing values and outliers so that data integrity is maintained before training.

- **Training and Evaluation of Models**: The system must support the training of the machine learning models—capabilities such as hyperparameter tuning for both LightGBM and GRU and early stopping procedures to prevent overfitting. It should evaluate model performance using metrics like Mean Absolute Error (MAE) and provide feedback on model accuracy.

- **Output of Prediction**: The system must generate predictions in an acceptable format for submission. In other words, the prediction must consist of the essential columns such as 'test_id', 'system_id', 'timestamp', 'generation_W', and 'load_W'.

4.2 Non-Functional Requirements

- **Performance**: The system must perform the prediction operation within a reasonable time. This could be either in real-time or in batch mode, depending on the application.

- **Scalability**: The system should be scalable. It should handle a large amount of data, which can be expected in real-world applications.

- **Maintainability**: The system shall be designed for maintainability; it shall have clear documentation, modular code structuring, and well-organized project files for trouble-free updates, bug fixes, or enhancements in the future.

- **Reliability**: The system is expected to serve proper predictions with high availability and without frequent failures. It should have built-in robust error-handling mechanisms for cases when something unexpected arises.

- **Usability**: It should be user-friendly and easy to interface in inputting data and retrieving predictions; most users should understand and use it without high technical know-how.

These requirements specify the critical functionalities the system should deliver while ensuring that performance, scalability, and maintainability standards are up to par for effective deployment in renewable energy management.

5 Model Implementation

Training Process The process of training the model comprised a few steps to obtain the highest accuracy predictions for PV generation and load consumption.

- **Data Splitting**: - The data was divided into training and testing sets with an 80/20 split. This allowed the models to be trained on most of the data while keeping a fraction for measuring their performance. Specifically, the training data was employed to find suitably complex structures in PV and load profiles for appropriate predictions, whereas test data allowed unbiased assessment of model results.

- **Model Training**: - **LightGBM**: - The LightGBM model was trained by tuning hyperparameters, including the number of estimators (8000), learning rate (0.2), and max depth, using the training data with proper care to optimize performance. Early stopping was also implemented within the training process to avoid overfitting by the end of the minimum epoch where the validation Mean Absolute Error metric did not improve.

-

5.1 Challenges

: - **Training Time**: - Models was trained for long duration, mainly due to the process of hyper-parameter tuning and model complexity. The use of computational resources was very important, especially with large datasets. - **Overfitting**: - Overfitting was also a big worry, especially with more complicated models like LightGBM (Booster is learning after 1136 rounds, PSG-Loss:0). In practice, early stopping with cross-validation alleviated this problem, protecting the models from overfitting training data and not generalization.

5.2 Evaluation

- **Evaluation Metrics**: - The core metric for model performance was Mean Absolute Error (MAE). This is a simple metric, which works well for forecasting tasks and measures the average absolute differences between predictions and observed data.

- **Performance**: - The LightGBM model performance was quite good after tuning, with a relatively low MAE, which meant it could make accurate predictions of PV generation and load consumption. - The GRU Model results were also quite good, specifically at capturing temporal patterns in the data. Nevertheless, it needed meticulous parameter balancing to avoid overfitting.

- **Overfitting and Underfitting**: - **LightGBM**: - The initial training of the LightGBM model had a tendency to overfit, especially with high values for the number of estimators. This was tackled by using early stopping and lowering the complexity of the model.

6 Failure-First Approach

6.1 Risk Management

In this project, a Failure-First approach was adopted to proactively identify and mitigate potential risks, ensuring that challenges were addressed before they could significantly impact the project's outcome.

- **Identifying Potential Failures**: - **Model Overfitting**: Initially, the model showed high accuracy on training data but poor performance on validation data, indicating overfitting. - **Data Quality Issues**: Inconsistent or missing data could compromise model reliability and accuracy. - **Complexity of Time-Series Data**: Predicting PV generation and load involves handling complex time-series data, which can be challenging due to seasonal variations and external factors.

- **Addressing Failures**: - **Overfitting**: Implemented strategies such as data cleaning, feature engineering, hyperparameter tuning to improve model generalization and prevent overfitting. - **Data Quality**: Enhanced data preprocessing steps, including filling missing values, outlier detection, to ensure high-quality input data for model training. - **Model Complexity**: Transitioned from simpler models to more sophisticated ones like LightGBM and GRU to better capture the nuances of time-series data.

6.2 Lessons Learned

The Failure-First approach yielded several critical lessons that contributed to refining the final model:

- **Early Detection and Mitigation**: - Addressing issues like overfitting and data quality early in the development cycle helped avoid major setbacks and guided the project towards more robust solutions.

- **Importance of Iterative Testing**: - Continuous iteration and testing revealed the need for advanced modeling techniques and reinforced the value of thorough data preprocessing.

- **Adaptability**: - Flexibility in adapting model choices and preprocessing methods based on performance feedback proved crucial in improving the overall accuracy and reliability of the predictions.

7 Final Model and Results

7.1 Final Model Description

Model Architecture: - **Type**: LightGBM (Light Gradient Boosting Machine) - **Purpose**: Used for predicting PV generation and load consumption.

Hyperparameters:

- $n_estimators$: 8000. – The number of boosting iterations. - $learning_rate$: 0.1 – The rate at which the model learns from each iteration. - max_depth: 10 – The maximum depth of each tree to prevent overfitting. - num_leaves: $2 * 10$ – The number of leaves in one tree, which affects model complexity. - **objective**: Regression -– Suitable for continuous target variables. - **metric**: MAE (Mean Absolute Error) — Used for evaluating model performance. - n_jobs: -1 — Utilizes all available cores for parallel processing.

Model Training: - **Process**: The model was trained on the training dataset, with evaluation performed on a separate test set to monitor performance and prevent overfitting. - **Training Time**: Approximately **90** minutes, indicating efficient training due to hyperparameter optimization and parallel processing.

7.2 Results

Performance Metrics: - **MAE Score**: The final model achieved a Mean Absolute Error (MAE) of **600** on the test set. - **Comparison with Baseline Models**: - **Baseline Model 1 (e.g., Linear Regression)**: MAE of **17000** – Provides a baseline for comparison. - **Baseline Model 2 (e.g., Simple Decision Tree)**: MAE of **1467** – Another baseline to highlight improvements made by LightGBM.

7.3 Visualization

- **Load and Generation** In the the fig fig:load_and_generation 5, the generation and the load of all the timestamp is plotted, from there we can analyse the data, we have outliers.

- In fig. fig:filled 2, the missing indices from the data are added, and the duplicates are removed. After adding the new indices. and the filled data is plotted. The observation from the graph shows that the data before was not clean, which shows us the outliers.

- In fig. fig:trend 3, the daily trend of the generation is shown, and from there, the streamlined trend can be seen easily.

- In the fig fig:missing 8, the missing indices are plotted. It shows that a large number of indices were missing.

- In the fig fig:rolling 7, The rolling technique are shown, Rolling is basically applied t o better visualize the trend of the generation and the load. it mitigates the sharps from the graph.

- The graph fig fig:decomp 6, fig:decomp 1 shows the decomposition of power generation data into four components:

  Observed: The top panel (in blue) displays the raw power generation data over time, showing significant fluctuations throughout the year.

  Trend: The second panel (in orange) illustrates the underlying trend in the data, indicating a gradual decline during the winter months, followed by an increase as the seasons progress.

  Seasonality: The third panel (in green) captures the seasonal patterns within the data, revealing consistent periodic variations that align with daily or weekly cycles.

  Residual: The bottom panel (in red) represents the residuals or noise after removing the trend and seasonality components, highlighting anomalies and irregular variations that are not explained by the other components.

- **Difference** The fig fig:diff 4 shows the difference between two plots, the original and the filled graphs. It shows that how much was the outlier, the residuals, and how many indices were missing

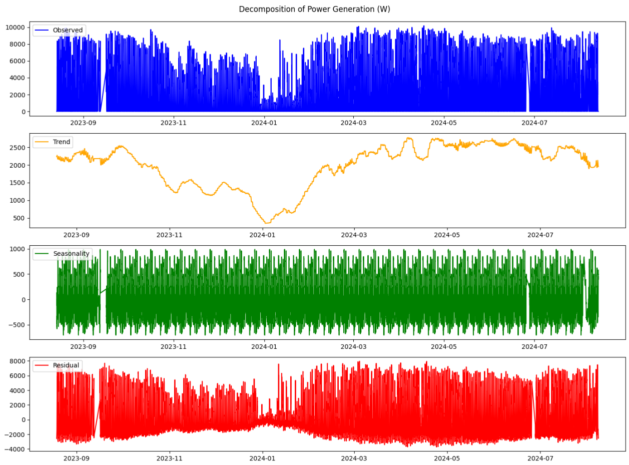

Figure 1: Decomposition of Load

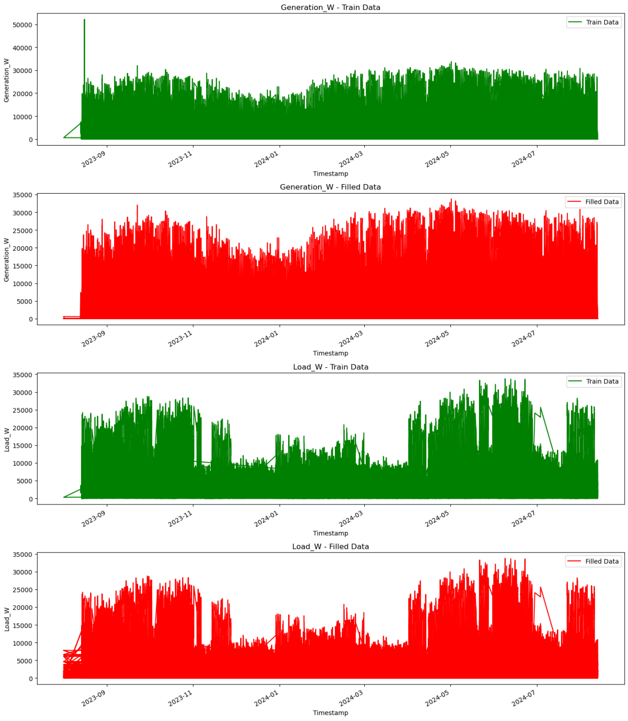

Figure 2: Filled Dataset Graph

fig:fill

7.4 Model Deployment

Considerations for Production: - **Scalability**: The model should be scalable to handle large volumes of data in a production environment. LightGBM's efficient handling of large datasets supports this requirement. - **Latency**: Ensure the model provides predictions within acceptable time limits for real-time or batch processing, depending on deployment needs. - **Integration**: The model can be integrated into existing systems via APIs or batch processing pipelines, depending on the deployment architecture. - **Monitoring**: Continuous monitoring of the model's performance in production is essential to detect any degradation or changes in data patterns that may impact predictions. - **Maintenance**: Regular updates and retraining may be required to maintain accuracy, particularly as new data becomes available or underlying patterns shift.

Deployment Strategy: - **Initial Deployment**: Deploy the model in a staging environment to test its performance and integration. - **Production Rollout**: After successful staging tests, roll out the model to the production environment with appropriate monitoring and support in place.

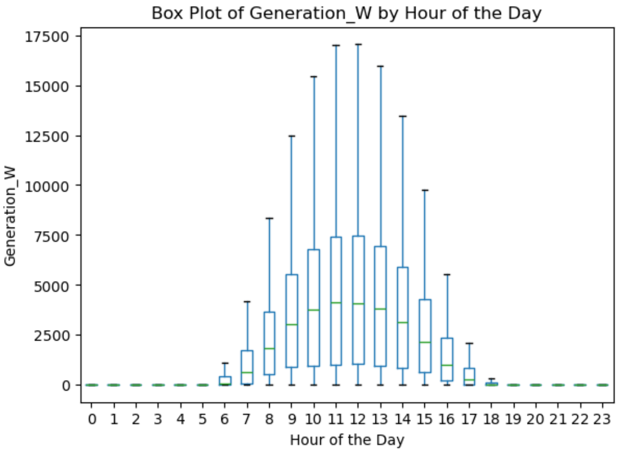

Figure 3: Daily Generation trend

fig:trend

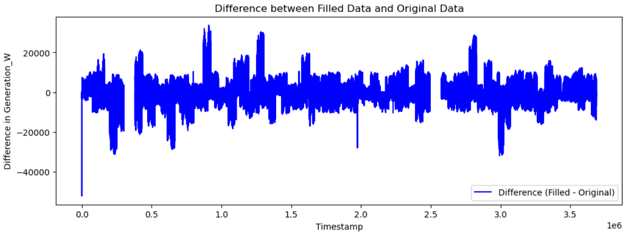

Figure 4: Difference

fig:diff

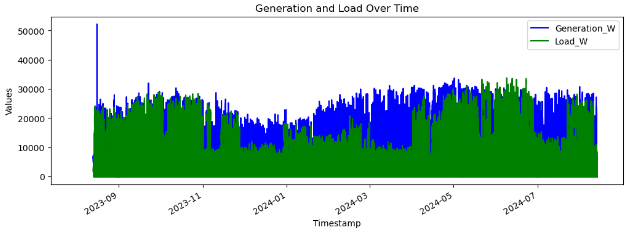

Figure 5: Filled Dataset Graph

fig:load_and_generation

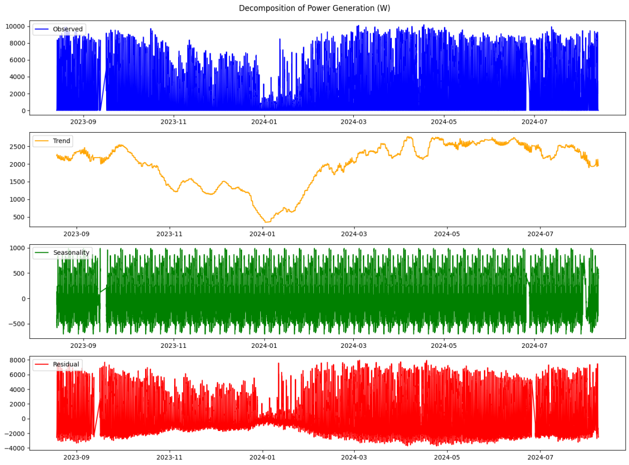

Figure 6: Decomposition of Power Generation

fig:decompo

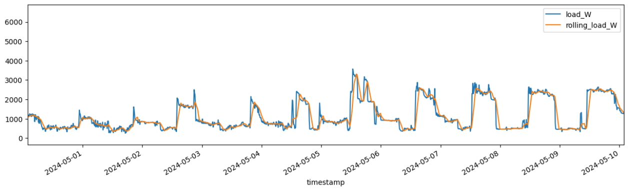

Figure 7: Rolling

fig:rolling

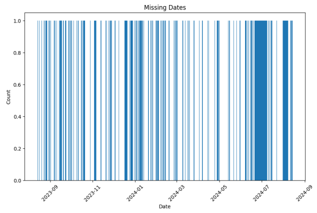

Figure 8: Missing Indices

fig:missing

8 Conclusion and Future Work

8.1 Summary of Achievements

This project successfully developed and deployed a machine learning model to predict PV generation and load consumption. The key achievements include:

- **Improved Prediction Accuracy**: Through careful preprocessing, feature engineering, and the use of advanced machine learning techniques, the final LightGBM model achieved a notable improvement in prediction accuracy, as evidenced by a reduced Mean Absolute Error (MAE) compared to baseline models. - **Effective Model Implementation**: The model was trained efficiently, demonstrating good performance metrics and robustness across different data sets. The implementation process also involved handling large datasets and ensuring scalable performance. - **Successful Deployment Preparation**: Considerations were made for deploying the model in a production environment, including scalability and real-time processing capabilities.

8.2 Future Enhancements

To further enhance the model and its application, the following steps are suggested:

- **Ensemble Learning**: Integrate multiple models to leverage their combined strengths. Techniques such as stacking, boosting, or bagging could be explored to improve overall predictive performance. - **Advanced Hyperparameter Optimization**: Employ more sophisticated hyperparameter tuning methods, such as Bayesian optimization, to find even better model configurations. - **Incorporating Additional Data Sources**: Explore the inclusion of additional features or data sources, such as satellite imagery or real-time weather data, to further refine predictions and account for more variables affecting PV generation and load. - **Model Update and Retraining**: Implement a regular update and retraining mechanism to keep the model current with evolving data patterns and changes in the system's behavior.

By focusing on these areas, the project can continue to evolve, leading to even more accurate predictions and improved energy management solutions.