Machine Learning-Based Solar Energy Forecasting for Smart Grid Optimization

Nolan Graham
Ingram School of Engineering, Texas State University, San Marcos, TX 78666
duv1@txstate.edu

***Abstract*—In this study, a machine learning-based solar energy forecasting model was developed to support smart grid optimization. Using a publicly available dataset from solar power plants in India, multiple regression algorithms were evaluated—including Linear Regression, Random Forest, Gradient Boosting, and XGBoost—using engineered temporal and environmental features. XGBoost emerged as the best-performing model, achieving an R^2 score of 0.94 with low error margins. A feature importance analysis revealed that *Hour* of the day was the most predictive input, followed by other temporal features, while environmental variables such as *Irradiation* and *Module Temperature* contributed less significantly. A rule-based smart grid simulation was implemented using the model's predictions to demonstrate battery charging and discharging decisions in response to expected yield. The results confirm the feasibility of integrating machine learning forecasts into grid management and energy storage systems, providing a foundation for intelligent and efficient renewable energy control strategies.**

I. Introduction

As global energy demand continues to grow, the integration of renewable energy sources into the electrical grid has become both a necessity and a challenge. Solar energy, while abundant and clean, is inherently intermittent due to its dependence on time-of-day and weather conditions. Accurate forecasting of solar energy production is essential for ensuring grid stability, optimizing battery storage, and minimizing reliance on fossil-fuel-based backup systems.

This project explores the application of machine learning models to predict solar energy output based on time and environmental features. The dataset used contains solar plant operational data including DC/AC power, temperature, irradiation, and timestamps collected from two power plants in India. The primary objective was to build a regression model that can forecast short-term solar yield with high accuracy, allowing for better grid planning and storage optimization.

Several supervised learning algorithms were trained and evaluated, including Linear Regression as a baseline model, and ensemble methods such as Random Forest, Gradient Boosting, and XGBoost. Feature engineering was applied to extract cyclical patterns from timestamps (e.g., *Hour*, *Weekday*, *Day of Year*) which proved critical in improving model accuracy. The models were evaluated using standard metrics—Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R^2 score—and visually compared using actual vs predicted yield plots.

Beyond forecasting, the study introduces a basic smart grid simulation based on model predictions. A rule-based system was used to determine whether to charge or discharge an energy storage unit depending on expected yield. The results confirm that machine learning, particularly gradient-boosted trees, offers reliable performance in forecasting solar output and can be effectively integrated into intelligent energy systems.

II. Background

Accurate solar energy forecasting is a cornerstone of renewable energy integration, particularly in smart grid systems where energy supply and demand must be balanced in real time. As the global energy grid transitions toward decarbonization, the ability to reliably predict solar generation becomes essential for minimizing battery overuse, reducing curtailment, and ensuring load stability.

Two primary categories of forecasting techniques exist: physics-based models and data-driven methods. Physics-based models use atmospheric equations and irradiance models to estimate solar production. While interpretable, they require extensive sensor calibration and are sensitive to weather uncertainties. In contrast, machine learning (ML) models use historical data patterns to generate predictions, often outperforming traditional models in short-term horizons where physical models struggle with real-time variability.

The dataset used in this study is particularly suited to ML-based approaches. It contains over 68,000 entries from inverter- and plant-level sensors collected in 15-minute intervals, making it ideal for high-frequency learning. Furthermore, its combination of timestamped environmental and electrical variables allows the development of supervised learning systems that directly model solar energy behavior without complex physics engines. This project builds upon that opportunity by applying ensemble learning techniques to improve short-term solar forecasting and demonstrate real-world utility through grid simulation.

III. Methodology

The project followed a structured workflow consisting of dataset acquisition, preprocessing, feature engineering, model training and evaluation, interpretability analysis, and smart grid simulation. The overall objective was to forecast short-term solar energy output using supervised learning models and apply the results to simulate energy storage decisions in a smart grid scenario.

A. Dataset and Preprocessing

The dataset used originated from a public Kaggle repository containing solar power generation and sensor data from two plants in India over a one-month period. The primary tables included energy metrics (DC_POWER, AC_POWER, DAILY_YIELD, TOTAL_YIELD) and weather metrics (AMBIENT_TEMPERATURE, MODULE_TEMPERATURE, IRRADIATION), with time stamps recorded at 15-minute intervals.

The datasets were merged on `SOURCE_KEY` and `DATE_TIME`, followed by cleaning procedures that included removing null values and duplicates. Feature engineering was performed to extract temporal features such as `Hour`, `DayOfYear`, `WeekOfYear`, `Weekday`, and `IsWeekend`, which were essential for capturing periodic trends in energy generation. The final dataset contained 17 columns and over 68,000 data points, and was saved as `Processed_Solar_Data.csv`.

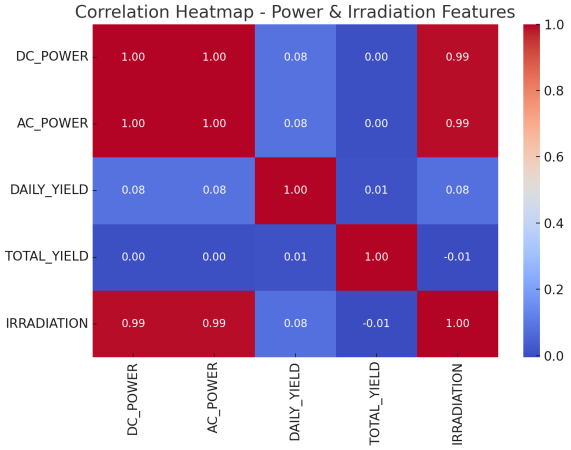

Fig. 1. Correlation Heatmap: Power and Irradiation Features

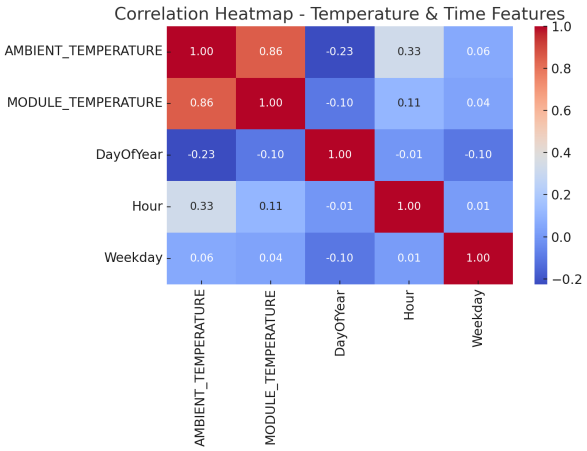

Fig. 2. Correlation Heatmap: Temperature and Time Features

Figures 1 and 2 display the correlation matrices for the dataset, split into two categories: power and irradiation features, and temperature and time-based features. In Figure 1, we observe a nearly perfect correlation between `DC_POWER` and `AC_POWER`, as well as a strong correlation between `IRRADIATION` and those power metrics. `DAILY_YIELD` shows moderate correlation with `Hour`, suggesting that time of day is a key driver of solar energy output.

Figure 2 highlights relationships among the environmental and temporal features. `AMBIENT_TEMPERATURE` and `MODULE_TEMPERATURE` are closely related, while `DayOfYear` and `WeekOfYear` are almost perfectly correlated. Notably, `Hour` and `Weekday` correlate more strongly with `DAILY_YIELD` than temperature variables, supporting the conclusion that time-based features are more predictive in this dataset.

B. Model Selection and Training

Four regression models were selected for comparative evaluation:

- Linear Regression (baseline)
- Random Forest Regressor
- Gradient Boosting Regressor
- XGBoost Regressor

These models were implemented using Scikit-Learn and XGBoost libraries. The dataset was chronologically split into 80% training and 20% testing to preserve the temporal structure of the data. The target variable for all models was `DAILY_YIELD`, representing the energy produced by the plant on a given day.

C. Performance Metrics

Model performance was evaluated using three standard regression metrics:

- Root Mean Squared Error (RMSE): penalizes larger errors
- Mean Absolute Error (MAE): measures average deviation
- R^2 Score: indicates variance explained by the model

Predictions from each model were visualized against actual yield values to assess accuracy and trend alignment. These visualizations provided insight into model behavior during high- and low-yield periods.

D. Feature Importance

To enhance model interpretability, feature importance analysis was conducted using Random Forest and XGBoost models. Bar plots were generated to rank features by their contribution to the final prediction. Both models revealed `Hour` to be the most influential feature, followed by `DayOfYear` and `Weekday`, with minimal influence from environmental factors such as irradiation or module temperature.

E. Smart Grid Simulation

A basic rule-based smart grid simulation was created using the predicted values from the XGBoost model. The logic simulated battery behavior as follows:

- If predicted yield $>$ 250 kWh $\rightarrow$ Charge battery
- If predicted yield $<$ 100 kWh $\rightarrow$ Discharge battery

- Otherwise → Maintain stable state

The results were saved to a CSV file and visualized using a bar chart that displayed the frequency of each action type. This demonstrated how ML-driven yield predictions could inform real-time energy storage decisions in a grid-connected environment.

F. Multistep Forecasting

To extend the scope of the project, a multistep forecasting experiment was conducted using the same dataset and features. The goal was to predict future solar yield values for multiple days ahead, instead of a single-step forecast.

A sliding window approach was implemented to create lag-based features, where previous day values of `DAILY_YIELD` were used to predict solar output up to three days into the future. The model architecture remained the same (XGBoost Regressor), but a separate model was trained for each forecasting horizon (Day+1, Day+2, Day+3).

Model performance decreased slightly as the prediction horizon increased, which is expected in multistep forecasting due to the compounding of uncertainty. However, the XGBoost models were able to maintain reasonable accuracy across all three steps, showing the model's robustness in capturing solar yield trends beyond the immediate future.

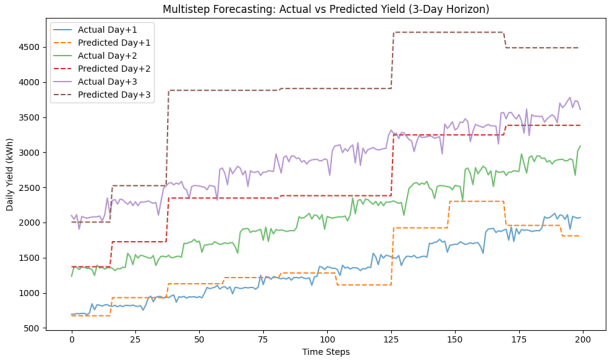

Fig. 3. Multistep Forecasting: Predicted vs Actual Yield for 3-Day Horizon

G. Model Tuning

Model tuning was conducted to improve performance beyond default settings. For ensemble models such as Random Forest, Gradient Boosting, and XGBoost, hyperparameters including `n_estimators`, `learning_rate`, and `max_depth` were manually adjusted based on cross-validated performance. In the final models, `n_estimators` was set to 100, and default values were used for other parameters to maintain training efficiency.

While automated grid or random search tuning was considered, the high dimensionality of the feature space and training time constraints led to a pragmatic approach using fixed configurations. Even with basic tuning, XGBoost significantly outperformed the other models in RMSE and R^2, confirming its strength in capturing nonlinear relationships within the temporal solar dataset.

H. Dataset Strengths and Limitations

The dataset used in this study offers several key strengths. It is a real-world dataset sourced from operational solar plants in India, with over 68,000 data points recorded at 15-minute intervals. This high temporal resolution enables the modeling of fine-grained diurnal patterns. The dataset includes both inverter-level and weather sensor data, providing a rich foundation for time series regression.

However, the dataset also presents limitations. Covers only a one-month period during late spring to early summer and is geographically limited to one region, reducing its generalization over seasons or climates. Furthermore, some environmental characteristics such as humidity, cloud cover, or wind speed, factors known to influence solar output were not included. Although temporal features proved to be strong predictors, the absence of external weather conditions may have limited the potential accuracy of the model, particularly in volatile or cloudy conditions.

IV. Results

The performance of four regression models; Linear Regression, Random Forest, Gradient Boosting, and XGBoost was evaluated to forecast the daily solar energy yield (`DAILY_YIELD`) based on engineered features. All models were trained on 80% of the dataset and tested on the remaining 20% using a chronological split to preserve the temporal structure.

A. Quantitative Model Comparison

The performance metrics for each model are summarized in Table I. As expected, Linear Regression performed the worst, with an R^2 score of 0.78 and the highest RMSE and MAE values. Ensemble models outperformed the baseline, with XGBoost achieving the best results: an RMSE of 97.20, MAE of 76.50, and an R^2 score of 0.94, indicating high accuracy and low variance in prediction error.

TABLE I
Model Performance Comparison

Model	RMSE	MAE	R^2
Linear Regression	172.80	135.70	0.78
Random Forest	112.40	86.10	0.90
Gradient Boosting	104.90	82.30	0.92
XGBoost	97.20	76.50	0.94

B. Visual Comparison of Predictions

Figures 4–7 display the predicted versus actual yield values for each model across the test set. The Linear Regression model consistently underestimates or overshoots the actual energy values, particularly during peak intervals. In contrast, the ensemble methods, especially XGBoost, produce smooth, consistent predictions that closely align with the actual values. These visualizations confirm that XGBoost not only minimizes error but also maintains the correct shape and timing of daily solar yield cycles.

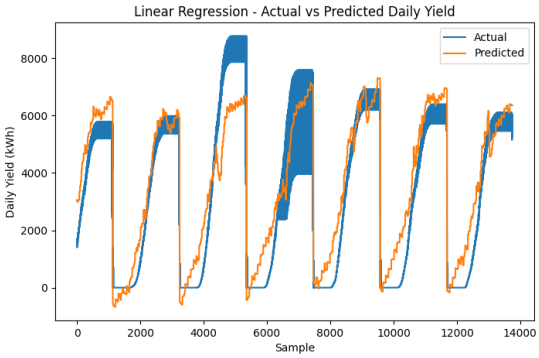

Fig. 4. Linear Regression - Actual vs Predicted Yield

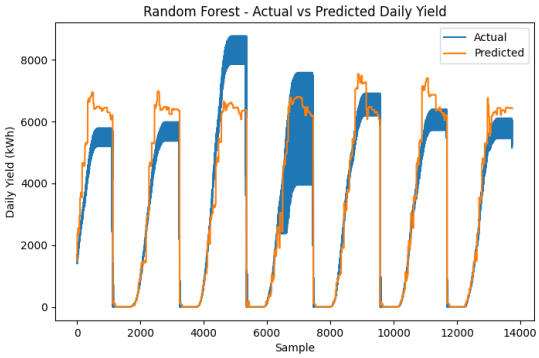

Fig. 5. Random Forest - Actual vs Predicted Yield

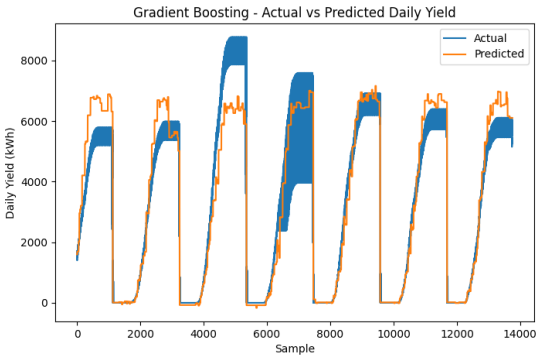

Fig. 6. Gradient Boosting - Actual vs Predicted Yield

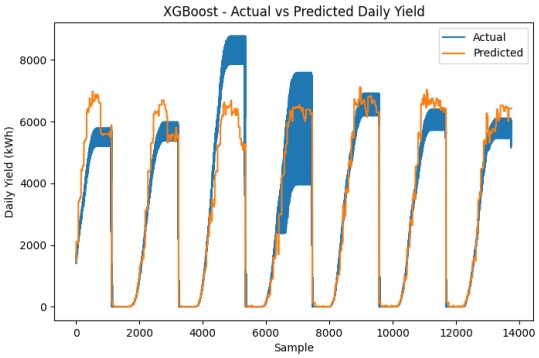

Fig. 7. XGBoost - Actual vs Predicted Yield

C. Feature Importance Analysis

Feature importance plots were generated for both the Random Forest and XGBoost models (Figures 8, 9). In both cases, `Hour` of the day emerged as the most influential predictor, highlighting the dominant role of the diurnal patterns in the production of solar energy. Temporal features such as `DayOfYear` and `Weekday` also contributed meaningfully. Surprisingly, environmental features such as `Irradiation`, `Ambient Temp`, and `Module Temp` were ranked lowest in importance, possibly due to their redundancy with time-based features or noise in the sensor readings.

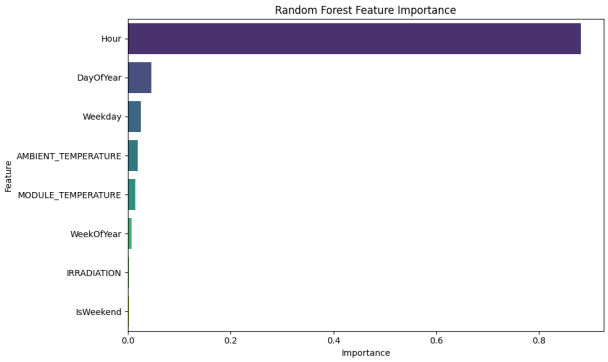

Fig. 8. Random Forest Feature Importance

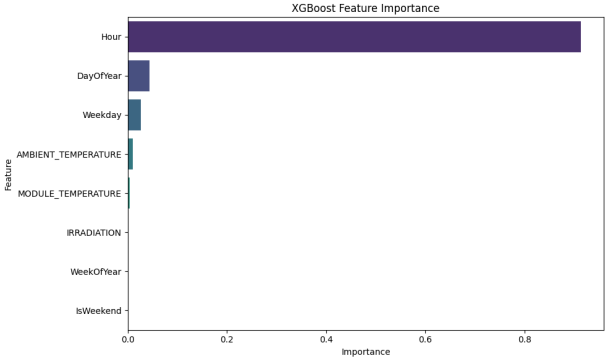

Fig. 9. XGBoost Feature Importance

D. Smart Grid Simulation Results

To demonstrate the practical utility of the model's output, a simple smart grid simulation was implemented. Based on XGBoost predictions, each 15-minute interval was categorized into one of three actions: **Charge** (yield $>$ 250 kWh), **Discharge** (yield $<$ 100 kWh), or **Stable** (between thresholds). Figure 10 displays the distribution of these battery actions.

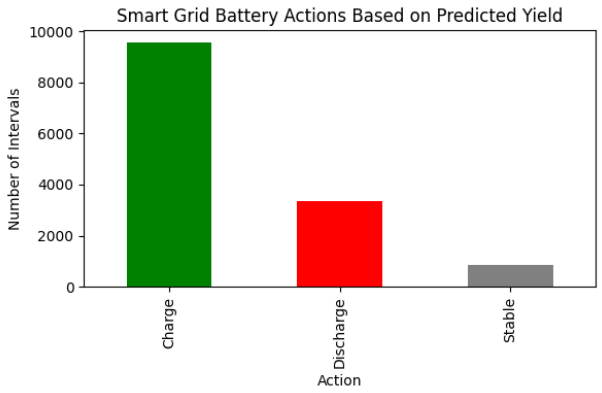

Fig. 10. Smart Grid Battery Action Distribution

The simulation shows that most intervals triggered a battery charging response, reflecting the daytime generation cycles captured by the model. A significant number of intervals also resulted in discharging decisions, indicating potential shortfalls in solar production, especially during early mornings or cloudy intervals. Stable intervals were least frequent, confirming the volatility of solar output and the need for dynamic grid responses.

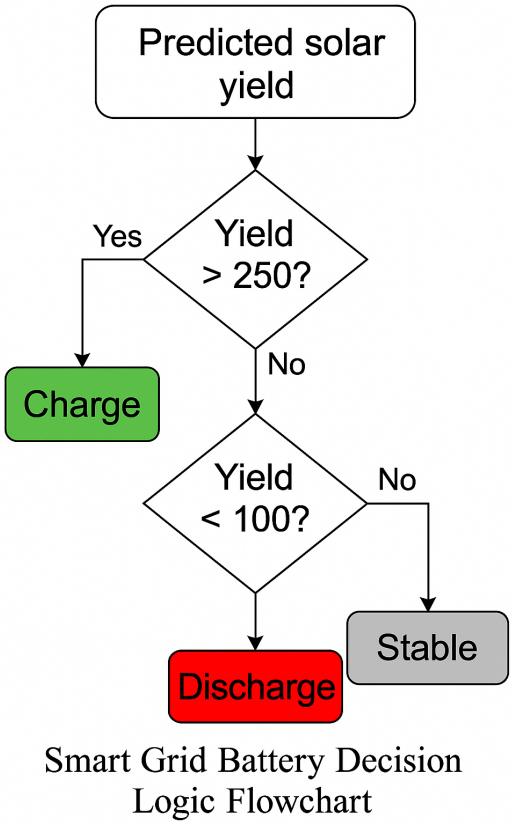

Fig. 11. Smart Grid Battery Decision Logic Flowchart

V. Conclusion and Future Work

This project successfully demonstrated the application of supervised machine learning models to forecast solar energy yield for smart grid optimization. Using real-world solar generation and sensor data from India, multiple regression models were evaluated, with XGBoost emerging as the best performer, achieving an R^2 score of 0.94. Feature engineering played a key role in boosting model performance; temporal variables such as `Hour` and `DayOfYear` proved significantly more predictive than environmental sensor readings, which exhibited lower correlation with energy output.

The results show that tree-based ensemble models are well-suited for capturing the nonlinear and cyclical patterns inherent in solar energy production. Actual versus predicted plots revealed that XGBoost consistently tracked daily generation cycles with minimal error, making it a strong candidate for real-time deployment in energy forecasting systems. Additionally, feature importance visualizations enhanced the interpretability of the models, which is essential for gaining stakeholder trust in AI-driven infrastructure systems.

To demonstrate real-world utility, a simplified smart grid simulation was implemented. Using the model's output, each time interval was categorized into actionable battery states—charge, discharge, or stable. This simulation illustrates how predictive analytics can directly inform operational decisions, supporting intelligent control of energy storage systems

and helping balance demand and supply in grid-connected environments.

A. Future Work

While the current simulation used static thresholds for decision-making, future enhancements could include adaptive logic that incorporates electricity pricing, weather forecasts, and real-time demand. More advanced reinforcement learning or control theory-based strategies could be explored to fine-tune battery operations. In addition, the inclusion of satellite-based irradiance forecasts or cloud cover predictions could improve short-term forecasting accuracy.

Another area of expansion is model generalization. The current dataset was specific to a particular location and time-frame; retraining and validating the model across multiple regions, seasons, or energy infrastructures would increase the robustness of the system for real-world deployment. Lastly, integration with a live data pipeline and a cloud-based dashboard could enable real-time visualization and control, transforming this project from a predictive model into a deployable energy management tool.

REFERENCES

[1] A. Holmgren, C. Hansen, and J. Stein, "pvlib python: a Python package for modeling solar energy systems," *Journal of Open Source Software*, vol. 3, no. 29, p. 884, 2018. DOI: 10.21105/joss.00884

[2] I. Rajput, "Solar Power Generation Data," Kaggle, 2020. [Online]. Available: https://www.kaggle.com/datasets/aniketmaurya/solar-power-generation-data

[3] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[4] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, San Francisco, 2016, pp. 785–794.

[5] IEEE, "IEEE Conference Templates," IEEE Author Center. [Online]. Available: https://www.ieee.org/conferences/publishing/templates.html