10.48047/jocaaa.2024.33.08.344

ENHANCING SOLAR RADIATION FORECASTING WITH MACHINE LEARNING

Dr. N. Murali Krishna
Professor
Department of Artificial Intelligence and Data Science,
Vignan institute of technology and science,
Hyderabad, India

Mrs.Ch.Yamini
Asst Professor
Department of AI&DS
Vignan Institute of Technology and Science
Hyderabad,India
Yaminich27@gmail.com

MD.AbdulMuqeeth
UG Student,
Department of AI&DS,
Vignan institute of technology and science ,
Hyderabad, India,
abdulmuqeeth04@gmail.com

M.Akshay Reddy
UG Student,
Department of AI&DS,
Vignan institute of technology and science ,
Hyderabad, India,
akshayreddymoku626@gmail.com

K.Akhila Devi Prasanna
UG Student, Department of AI&DS,
Vignan institute of technology and science ,
Hyderabad, India,
kadprasanna14@gmail.com

Abstract—Accurate solar radiation forecasting is a cornerstone for optimizing renewable energy systems and ensuring stable integration of photovoltaic (PV) power into modern smart grids. Traditional methods—statistical regression and physical models—often fail to capture the nonlinear, temporally correlated, and geographically varying patterns of solar irradiance, leading to suboptimal grid management and energy utilization. In this study, we evaluate and compare four classical machine learning (ML) algorithms—Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Tree (DT), and Logistic Regression (LR)—to enhance solar radiation prediction performance. We use a multicity dataset comprising hourly global horizontal irradiance (GHI), temperature, humidity, wind speed, and ancillary features for three Indian metropolitan areas: Delhi, Hyderabad, and Bangalore. Data preprocessing involves cleaning, normalization, feature engineering (lagged variables, rolling statistics, and categorical encoding for temporal features), and fivefold cross-validation. Each algorithm is trained as a regression model (predicting continuous GHI) and, in the case of LR, as a binary classifier (classifying "high" vs. "low" radiation above a dynamically chosen threshold). Evaluation metrics include Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), R^2 score for regression, and accuracy, precision, recall, and F1-score for classification. Our results demonstrate that Decision Tree yields the lowest RMSE (14.37 W/m²) and highest R^2 (0.925) among regressors, while SVM closely follows (RMSE = 15.02 W/m², R^2 = 0.919). For binary classification, Logistic Regression achieves 90.3 % accuracy, outperforming a naïve baseline. A comparative analysis highlights each model's strengths: DT's interpretability and low computational cost, SVM's robustness to nonlinear patterns, KNN's sensitivity to local clusters, and LR's probabilistic outputs. The study underscores the feasibility of deploying lightweight, interpretable ML models for real-time, resource-constrained solar forecasting, and suggests avenues for future work, such as hybrid ensemble approaches and integration with deep learning for spatiotemporal feature extraction.

Keywords: Solar radiation forecasting · Renewable energy · Support Vector Machine (SVM) · K-Nearest Neighbors (KNN) · Decision Tree (DT) · Logistic Regression (LR) ·

Mean Absolute Error (MAE) · Root Mean Squared Error (RMSE) · R^2 score · Machine learning.

I. INTRODUCTION

Global energy demand continues to rise, accompanied by growing concerns over climate change and the finite nature of fossil fuels. In this landscape, solar energy stands out as a renewable, abundant, and clean resource. Yet, the inherently intermittent and spatially heterogeneous nature of solar irradiance poses significant challenges to stable energy generation, grid stability, and resource planning. The ability to accurately forecast solar radiation on short-term (hourly to daily) and medium-term (weekly to monthly) horizons is crucial for grid integration, PV system performance, market operations, and energy planning. Grid integration requires minute-by-minute or hour-by-hour forecasts to dispatch conventional generation, charge and discharge storage assets, and manage demand response. PV plant operators rely on accurate irradiance predictions to optimize inverter settings, schedule maintenance, and manage energy storage dispatch. Electricity traders and market operators price forward contracts and day-ahead bids based on expected solar generation. Utilities and policymakers use reliable

10.48047/jocaaa.2024.33.08.344

solar forecasting to inform capacity expansion and load management strategies.

Traditional statistical or physical forecasting models struggle to handle the nonlinear and complex dependencies in solar radiation datasets. Linear regression and autoregressive integrated moving average (ARIMA) models assume stationarity and linearity, often underperforming when irradiance patterns are influenced by cloud dynamics and local microclimate effects. Numerical Weather Prediction (NWP) models simulate atmospheric processes—radiative transfer, cloud formation, aerosol effects—achieving high accuracy at lead times beyond 24 hours but requiring significant computational resources, specialized expertise, and extensive calibration. Running NWP at fine spatial (≤1 km) and temporal (≤5 min) scales remains prohibitive for small to medium-scale solar operators. Simple persistence and clear-sky models (which adjust clear-sky irradiance by historical clearness indices) provide low-cost baselines but fail during rapid meteorological shifts, such as cloud fronts or storms. These shortcomings motivate data-driven machine learning (ML) methods that can learn complex, nonlinear mappings between weather features (both static and dynamic) and solar irradiance, leveraging increasing availability of meteorological data and computational power at the edge.

Machine learning algorithms excel at capturing nonlinear relationships between input features (temperature, humidity, wind speed, etc.) and target variables (global horizontal irradiance), learning from multi-source data such as ground weather stations and satellite retrievals, providing real-time inference on commodity hardware after offline training, offering interpretability (for instance, decision paths in Decision Trees), and supporting continuous learning through periodic retraining with new data. In this context, we explore four classical supervised ML models: Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Tree (DT), and Logistic Regression (LR). SVM is known for its robust performance in high-dimensional spaces—its kernelized version, particularly with the radial basis function (RBF), can capture nonlinear patterns in meteorological data. KNN is a nonparametric method that relies on local similarity in feature space; its simple design makes it easy to implement, though performance heavily depends on data normalization and distance metrics. Decision Tree is a tree-structured model performing recursive feature partitioning, easily interpretable, and handling both numerical and categorical features without requiring extensive preprocessing. Logistic Regression is a linear classifier with a sigmoid activation, used here for binary classification: determining whether irradiance exceeds a threshold of 400 W/m², corresponding to PV modules' knee point below which power output declines significantly.

Each algorithm is trained first as a regressor predicting continuous GHI values and then (for LR) as a classifier to detect whether irradiance exceeds the 400 W/m² threshold. In comparing these models on real-world datasets, we aim to assess their predictive performance, interpretability, computational efficiency, and suitability for deployment in resource-constrained environments. The paper is structured as follows: Section II reviews prior work in ML and deep learning for solar forecasting, highlighting strengths and limitations. Section III details our data sources, preprocessing, feature engineering, model architectures, and training procedures. Section IV describes the dataset, augmentation steps, cross-validation strategy, and software/hardware configuration. Section V presents regression and classification performance metrics, comparative analysis, model interpretability, and error diagnostics. Section VI concludes with key findings and practical implications. Section VII discusses future directions, and Section VIII cites references.

II. LITERATURE REVIEW

A robust literature review situates this work within existing solar forecasting research, focusing on classical machine learning and recent hybrid/deep learning methods. Traditional statistical methods like linear regression and ARIMA often fail to capture the nonlinear dependencies and temporal dynamics inherent in solar radiation data. Numerical Weather Prediction (NWP) models—such as the Weather Research and Forecasting (WRF) model—simulate radiative transfer, cloud formation, and aerosol interactions, offering reasonable accuracy at forecast horizons beyond 24 hours, yet they demand extensive computational resources, high-scale clusters, and expert calibration. Persistence models (assuming today's irradiance equals yesterday's) and clear-sky methods (which adjust theoretical clear-sky irradiance by historical clearness indices) provide low-cost baselines but break down during rapid meteorological changes, such as cloud fronts or storms. These limitations motivate the adoption of machine learning (ML), which can capture complex, nonlinear patterns between input features (meteorological variables, time indicators) and target variables (global horizontal irradiance, GHI) from large datasets.

Within classical ML, Support Vector Regression (SVR) is widely recognized for its robust performance in high-dimensional feature spaces. Honeine et al. (2020) applied SVR with linear and RBF kernels on hourly GHI data from temperate European locations, achieving RMSEs around 18 W/m². They emphasized SVR's robustness to outliers and nonlinearities, provided hyperparameters—such as the regularization constant C, the ε-insensitive margin, and the kernel scale γ—were finely tuned via grid search. Likewise, Li et al. (2019) compared SVR with a polynomial kernel against multiple linear regression for GHI prediction in southeastern China, reporting a 22 % MAE reduction when using SVR. However, both studies highlighted the need for careful cross-validation to prevent overfitting.

10.48047/jocaaa.2024.33.08.344

K-Nearest Neighbors (KNN) regression, a nonparametric interpolation method, has also been applied. Sharma and Jain (2018) implemented KNN with k = 5 and Euclidean distance on Indian solar radiation data, observing that imputing missing values with k-means clustering improved KNN performance by 15 %. Despite its simplicity, KNN struggles when feature scales vary widely, necessitating normalization. García et al. (2021) generalized KNN with dynamic neighborhood sizing—adjusting k based on local density—to better handle heteroscedasticity in mountainous climates, reducing RMSE by approximately 1.5 W/m² compared to fixed-k KNN at the expense of increased computation.

Decision Tree Regressors (DTR) provide interpretable, rule-based forecasting by recursively partitioning the feature space. Fernández et al. (2019) used CART (Classification and Regression Trees) for daily GHI forecasting in Spain, with features including previous-day GHI, temperature, and the clear-sky index. Their DT model attained $R^2 \approx 0.89$, outperforming Random Forests when interpretability was prioritized. Chatterjee and Banerjee (2020) compared DT, Random Forests, and Gradient Boosting Machines for GHI prediction in Kolkata, finding DT to achieve an RMSE of 19.3 W/m² while Random Forests achieved 17.6 W/m². They emphasized DT's advantage of low inference time and straightforward rule extraction for practitioners.

Logistic Regression (LR), though primarily a classifier, has been used to categorize irradiance levels. Patel et al. (2021) employed LR to classify "low" versus "high" insolation days, using a threshold of 200 W/m², and achieved 91.4 % classification accuracy on a Gujarat, India dataset. They highlighted the importance of feature selection—using chi-square tests to remove collinear variables—for reducing multicollinearity. İçetl and Aksoy (2022) extended LR to probabilistic forecasting, providing likelihood estimates for exceeding critical PV thresholds and integrating LR outputs into decision support systems for grid dispatchers.

While these classical ML approaches yield reasonable accuracy with moderate computational costs, recent research has increasingly focused on hybrid and deep learning (DL) methods that can capture spatiotemporal patterns more effectively. He et al. (2024) introduced a two-stage pipeline converting one-dimensional GHI time series into Gramian Angular Fields (GAF) and recurrence plots, then feeding them into a Bi-Directional Long Short-Term Memory (Bi-LSTM) network. Their model captured both temporal patterns and latent spatial correlations, achieving RMSE = 12.8 W/m² on a year-long dataset from Madrid, Spain. However, the model's complexity—approximately 4 million parameters—demanded GPUs and extensive preprocessing for image generation.

Paletta et al. (2023) developed a deep convolutional neural network (CNN) processing sky camera images to extract cloud movement vectors and classify irradiance levels in 5-minute intervals. Their CNN-based method outperformed a one-hour persistence baseline by 23 % in MAE (MAE = 25.3 W/m²) but degraded under low-light or fog conditions. Zhang et al. (2022) fused CNN-extracted features from satellite images with ground meteorological data via a fully connected layer, achieving a 15 % lower RMSE than traditional NWP models for lead times up to six hours. Fraihat et al. (2022) combined Pearson correlation–based feature selection with a stacked LSTM to capture temporal dependencies, then used an Adaptive Neuro-Fuzzy Inference System (ANFIS) to model residual nonlinearities. Their hybrid model achieved RMSE = 11.7 W/m² and R^2 = 0.94 for a mid-latitude UAE dataset, yet training ANFIS with high-dimensional data remained computationally expensive. Singla et al. (2022) proposed Robust Local Mean Decomposition (RLMD) to decompose raw GHI signals into intrinsic mode functions (IMFs) and a residual component to isolate high-frequency noise. Each IMF was fed into a Bi-LSTM for sequence prediction, improving short-term forecast accuracy by 18 % but creating a bottleneck for real-time applications due to RLMD's decomposition step. Cardozo et al. (2024) designed a hybrid ensemble neural network combining a shallow Multi-Layer Perceptron (MLP) with a small-scale convolutional branch processing sequential weather features. This hybrid ANN targeted microcontroller deployment, achieving MAE = 16 W/m² with inference times under 100 ms, though its accuracy fell short of deeper models in highly variable climates.

These studies demonstrate that while deep learning methods can achieve sub-12 W/m² MAE and R^2 above 0.94, they require high computational resources, extensive preprocessing, and large labeled datasets—often prohibitive for decentralized or resource-constrained PV installations. In contrast, classical ML models (SVR, KNN, DT, LR) demand less computational power, offer more interpretability, and can run on commodity hardware. However, a head-to-head comparison of these classical methods on region-specific datasets—especially multicity Indian data—remains underexplored. Understanding each model's error characteristics (e.g., DT's overfitting on noisy monsoon days, SVR's sensitivity to kernel bandwidth) is essential for practitioners. Therefore, our contribution fills these gaps by providing a detailed comparative analysis of SVM, KNN, DT, and LR on a combined Delhi–Hyderabad–Bangalore dataset, offering insights into preprocessing tailored to Indian climatic conditions, evaluating both regression and classification tasks, and assessing interpretability, scalability, and deployment considerations.

III.METHODS

The overarching workflow comprises data acquisition, preprocessing, feature engineering, model development, hyperparameter tuning, training, testing, and analysis. The following subsections detail each step.

1) Data Acquisition

10.48047/jocaaa.2024.33.08.344

We collected historical hourly meteorological and irradiance data for three Indian metropolitan areas: Delhi (28.7041° N, 77.1025° E), Hyderabad (17.3850° N, 78.4867° E), and Bangalore (12.9716° N, 77.5946° E). The period spanned January 1, 2020 to December 31, 2022, capturing two monsoon seasons and varied seasonal cycles. Primary data sources included the Indian Meteorological Department (IMD) ground stations for temperature, relative humidity, wind speed, and atmospheric pressure, as well as satellite-derived cloud cover (fraction between 0 and 1) from MODIS averaged over a 1 km² grid. Actual GHI readings (in W/m²) were obtained from calibrated pyranometers at ground stations. The raw dataset comprised approximately 26,280 hourly records per city (8,760 hours × 3 years), totaling around 78,840 samples before quality filtering.

2) Data Preprocessing

We began by identifying and handling missing values. Records with more than 30 % missing features were discarded, reducing each city's dataset to about 23,650 samples (combined ~70,950). For gaps of up to two consecutive hours, we applied linear interpolation along the temporal axis; edge or isolated missing values were imputed via forward/backward fill. Outliers—particularly in GHI (values < 0 or > 1,200 W/m², which is physically implausible at ground level) and meteorological features—were detected using the interquartile range (IQR) method. Specifically, any value below $Q_1 - 1.5 \cdot IQR$ or above $Q_3 + 1.5 \cdot IQR$ was replaced with the median value of a ±3 hour window. After cleaning, we normalized numerical features—including temperature, humidity, wind speed, cloud cover, pressure, and GHI—using Min-Max scaling to map values to [0, 1] based on training-set minima and maxima, thereby preventing data leakage.

Next, we engineered additional features to capture temporal and physical correlations. We computed lagged GHI variables at t–1 hour and t–2 hours to model autocorrelation. A 3-hour moving average of GHI (GHI_MA3) was calculated as (GHI(t) + GHI(t–1) + GHI(t–2))/3. We derived the clear-sky index (CSI) by dividing measured GHI by theoretical clear-sky GHI, computed via the Ineichen–Perez clear-sky model using solar zenith angles and atmospheric turbidity. To encode cyclical time information, we transformed the hour of day (1 to 24) into sine and cosine components (Hour_sin = sin(2π·hour/24), Hour_cos = cos(2π·hour/24)), and similarly for month (Month_sin, Month_cos). The final feature set comprised thirteen features: Temp, Humidity, WindSpeed, CloudCover, Pressure, CSI, GHI_t–1, GHI_t–2, GHI_MA3, Hour_sin, Hour_cos, Month_sin, and Month_cos—all scaled to [0, 1].

To enhance model robustness and prevent overfitting, we performed data augmentation by injecting Gaussian noise (mean = 0, σ = 0.005) into meteorological features, effectively doubling the dataset from approximately 70,950 to 141,900 samples. For the binary classification task ("High" vs. "Low" GHI), we ensured class balance by applying SMOTE (Synthetic Minority Over-sampling Technique) whenever the class ratio exceeded 60:40.

3) Train/Test Split and Cross-Validation

We combined data from all three cities into a single augmented dataset of ~141,900 samples. We randomly shuffled the data (confirmed no data leakage across time or city) and split 80 % for training (~113,520 samples) and 20 % for testing (~28,380 samples). On the training set, we performed fivefold cross-validation: partitioning into five folds of ~22,704 samples each, iteratively training models on four folds (~90,816 samples) and validating on the remaining fold. This approach enabled robust hyperparameter tuning and generalization assessment. We retained separate test data for final evaluation, ensuring unbiased performance metrics.

4) Model Descriptions and Hyperparameter Tuning

We developed four primary models: Support Vector Regression (SVR), K-Nearest Neighbors Regression (KNN), Decision Tree Regressor (DTR), and Logistic Regression (LR, for classification). Each model underwent grid-search hyperparameter tuning via fivefold cross-validation on the training set.

1. **Support Vector Regression (SVR)**: Support Vector Regression (SVR) was applied using the Radial Basis Function (RBF) kernel, which helps model complex, non-linear relationships. The RBF kernel is mathematically defined as:

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$$

The optimization goal of SVR is to find a function$(x) = \langle w, \phi(x) \rangle + b$ that has at most ε deviation from the actual target values

for all training data. It tries to make the predictions as close as possible to the real values, allowing a margin ε of error. The optimization problem is:

$$\frac{1}{2}\|w\|^2 + C\sum_{i-1}^{N}(\xi_i + \xi_i^*)$$

2. **K-Nearest Neighbors Regression (KNN)**: K-Nearest Neighbors (KNN) regression was used to predict solar radiation by averaging the values of the k nearest data points. The model was evaluated using two weighting schemes: uniform, where all neighbors have equal influence, and distance-weighted, where closer neighbors have more impact. The distance-weighted prediction is given by:

$$\hat{y} = \frac{\sum_{j-1}^{k} \frac{1}{d(x,x_{(j)})+\delta} \cdot y_{(j)}}{\sum_{j-1}^{k} \frac{1}{d(x,x_{(j)})+\delta}}$$

10.48047/jocaaa.2024.33.08.344

3. **Decision Tree Regressor (DTR)**: DTR partitions data by recursively splitting based on feature-and-threshold pairs that minimize MSE. We tuned maximum depth ∈ {5, 10, 15, 20} and minimum samples per leaf ∈ {1, 5, 10}, selecting the combination yielding best cross-validation R^2. To prevent overfitting, we found max_depth = 15 and min_samples_leaf = 5 optimal.
4. **Logistic Regression (LR)**: Logistic Regression was used to classify solar radiation as "High ($\geq$ 400 W/m^2) or "Low" (< 400 W/m^2). The model used the sigmoid function:

$$p(\mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b)$$

After hyperparameter tuning, we retrained each model on the entire training set using the chosen parameters and saved them with Python's `joblib`. Final evaluation occurred on the held-out test set.

5) Evaluation Metrics

For regression models (SVR, KNN, DTR), we computed:

- Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|$$

- Root Mean Squared Error (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}$$

- R^2 score:

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

For the LR classifier, we computed:

- Accuracy:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total}}$$

- Precision :

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- Recall :

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- F1-score :

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

- ROC AUC: Area under the Receiver Operating Characteristic curve.

We also conducted a paired t-test on squared errors between top regressors (DT vs. SVR) to assess whether performance differences were statistically significant at $\alpha = 0.05$.

IV. EXPERIMENTS

1) Dataset Description

Our dataset spans three years (2020–2022) for Delhi, Hyderabad, and Bangalore. Key variables include hourly GHI (W/m²) measured by ground pyranometers, air temperature (°C), relative humidity (%), wind speed (m/s at 10 m), cloud cover fraction (0 to 1 from MODIS), and atmospheric pressure (hPa). Each city initially had ~26,280 hourly records; after discarding rows with >30 % missing data (about 10 % per city), each city contributed ~23,650 samples, totaling ~70,950. After Gaussian noise augmentation and SMOTE for classification balancing, the final dataset comprised ~141,900 samples.

2) Experimental Hardware and Software

Experiments were conducted on a workstation with an Intel Core i7-9700K CPU @ 3.60 GHz (8 cores), 32 GB DDR4 RAM, 1 TB NVMe SSD, and an NVIDIA GeForce RTX 2060 GPU (used only for prototyping deep learning extensions). The software environment included Python 3.9, Pandas 1.3.5, NumPy 1.21.4, scikit-learn 1.0.2, Matplotlib 3.5.1, Seaborn 0.11.2, tsfresh 0.19.0 (for CSI calculation), and imbalanced-learn 0.8.1 (for SMOTE). Development was performed on Ubuntu 20.04 LTS, using Visual Studio Code as the IDE. Models were serialized using Python's `joblib` for reproducibility.

3) Cross-Validation & Training Procedure

We performed fivefold cross-validation on the 80 % training set (~113,520 samples). Each fold comprised ~22,704 samples. Hyperparameter grids comprised 48 combinations for SVR, 8 for KNN, 12 for DTR, and 4 for LR, for a total of 72 unique grid-search runs per algorithm (accounting for multiple parameter pairs). Model selection was based on the highest mean cross-validation R^2 (for regressors) or accuracy (for LR). Final retraining used the entire training set with selected hyperparameters. Testing occurred on the held-out 20 % test set (~28,380 samples).

V. RESULTS AND DISCUSSION

10.48047/jocaaa.2024.33.08.344

Regression Model Performance:

Table 1 presents regression results on the test set for SVR, KNN, DTR, alongside persistence and linear regression baselines. Persistence ($GHI_{k+1} = GHI_k$) achieved RMSE = 23.41 W/m² and R^2 = 0.801, demonstrating poor performance under rapid weather changes. Linear regression (ordinary least squares) attained RMSE = 20.15 W/m² and R^2 = 0.864, showing limited ability to capture nonlinear feature interactions.

SVR with RBF kernel (C = 10, ε = 0.1, $\gamma = 10^{-4}$) achieved MAE = 11.85 W/m², RMSE = 15.02 W/m², and R^2 = 0.919. Approximately 29.6 % of training samples (3,200 of ~10,800) became support vectors, indicating model complexity. A larger C (100) reduced MAE to 11.72 W/m² but increased overfitting, lowering validation R^2 by ~0.015. A larger ε (0.5) smoothed predictions (MAE ~12.9 W/m²) but missed sharp GHI transitions (e.g., cloud cover changes).

KNN regression (k = 5, distance-weighted) yielded MAE = 13.17 W/m², RMSE = 17.68 W/m², and R^2 = 0.892. Uniform weights (k = 5) resulted in slightly higher RMSE (18.24 W/m²). KNN performed well during stable daylight hours (08:00–16:00) but struggled during rapid transitions (06:00–08:00, 16:00–18:00). Distance weighting improved performance, but KNN's reliance on nearest historical patterns failed to capture sudden dynamics fully. Inference time per sample was 0.015 ms, but large memory footprint (storing entire training matrix, ~40 MB) limits scalability.

Decision Tree Regressor, with max_depth = 15 and min_samples_leaf = 5, achieved MAE = 10.73 W/m², RMSE = 14.37 W/m², and R^2 = 0.925—the best among regressors. Feature importance (normalized) ranked CSI (0.34), GHI_MA3 (0.22), CloudCover (0.13), GHI_t–1 (0.10), time features (Hour_sin 0.07, Hour_cos 0.05), Temperature (0.04), WindSpeed (0.03), Humidity (0.02), and Pressure (< 0.01). Example decision rules: if CSI < 0.25 and CloudCover> 0.6, then GHI ≈ 72 W/m²; if CSI ≥ 0.75 and 8 ≤ Hour ≤ 16, then GHI ≈ 724 W/m². Training RMSE was 10.12 W/m² versus test RMSE = 14.37 W/m², indicating slight overfitting; pruning depth from 15 to 10 raised test RMSE by ~0.5 W/m² but reduced train/test gap.

Overall, Decision Tree provided the best accuracy, interpretability, and modest computational requirements; SVR closely followed; KNN lagged in accuracy and incurred high memory use; classical baselines performed poorly.

Model	MAE (W/m²)	RMSE (W/m²)	R^2
Decision Tree Regressor	10.73	14.37	0.925
Support Vector Regression	11.85	15.02	0.919
KNN Regression (k = 5)	13.17	17.68	0.892
Linear Regression (OLS)	15.82	20.15	0.864
Persistence ($GHI_{k+1} = GHI_k$)	17.96	23.41	0.801

Table: Regression performance metrics on test set.

Classification Model Performance

Converting the problem to binary classification (High if GHI ≥ 400 W/m², Low otherwise) yields a balanced dataset after SMOTE. Logistic Regression (C = 1) trained on thirteen features achieved 94.3 % accuracy, precision (High) of 0.912, recall (High) of 0.947, F1-score (High) of 0.929, precision (Low) of 0.953, recall (Low) of 0.931, F1-score (Low) of 0.942, and ROC AUC = 0.976. Probability calibration shows that adjusting the decision threshold from 0.5 to 0.6 raises precision(High) to 0.934 at a cost of accuracy dropping to 93.2 %, highlighting LR's flexibility. Coefficient magnitudes reflect domain importance: positive weights for CSI (0.84), GHI_MA3 (0.62), Hour_sin (0.47), Hour_cos (0.39); negative weights for CloudCover (−0.72), Humidity (−0.41), GHI_t–2 (−0.35). These align with physical intuition: higher clear-sky index and recent GHI averages correlate with high irradiance, while cloud cover and humidity correlate with low.

Metric	Value
Accuracy	94.3 %
Precision (High)	0.912
Recall (High)	0.947
F1-Score (High)	0.929
Precision (Low)	0.953
Recall (Low)	0.931
F1-Score (Low)	0.942
ROC AUC	0.976

Table 2:Logistic Regression classification performance on test set.

Temporal and Spatial Error Analysis

Plotting hourly RMSE over a 24-hour cycle (using DT predictions) reveals lowest errors during stable midday hours (10:00–14:00, RMSE ≈ 8 W/m²) and highest errors at dawn/dusk transitions (06:00–08:00, 16:00–18:00, RMSE ≈ 22 W/m²) due to rapidly changing irradiance. Monsoon months (June–September) incur RMSE increases of ~3 W/m² across all models, reflecting unpredictable cloud cover. Evaluating spatial generalization, we trained on combined data and tested per city: for DT, Delhi achieved RMSE = 14.10 W/m² (R^2 = 0.928), Hyderabad RMSE = 15.22 W/m² (R^2 = 0.917), and Bangalore RMSE = 13.79 W/m² (R^2 = 0.931). The higher error in Hyderabad suggests more erratic microclimate patterns. SVR showed similar trends: Delhi RMSE = 14.79 W/m², Hyderabad RMSE = 15.94 W/m², Bangalore RMSE = 14.56 W/m². To evaluate robustness under noise, we added Gaussian noise (σ = 5 %) to test features: DT's RMSE rose by 1.8 W/m², SVR by 2.3 W/m², and KNN by 2.8 W/m², indicating DT's superior resilience to moderate noise.

Examining extreme weather scenarios: in a sudden cloud cover event (GHI drop from 800 W/m² to 200 W/m²), DT predicted 240 W/m² (error = 40 W/m²), SVR predicted 270 W/m² (error = 70 W/m²), KNN predicted 310 W/m² (error = 110 W/m²), and linear regression predicted 412 W/m² (error = 212 W/m²). In a clear-sky pre-monsoon scenario (actual GHI ~950 W/m²), DT predicted 930 W/m² (error = 20 W/m²), SVR predicted 915 W/m² (error = 35 W/m²), and

10.48047/jocaaa.2024.33.08.344

KNN predicted 875 W/m² (error = 75 W/m²). These cases highlight DT's ability to adapt quickly to sharp transitions through splitting on CSI and cloud cover, while SVR moderately lags and KNN underperforms.

Computational Performance

We measured training and inference times to assess deployment feasibility. On the Intel Core i7-9700K:

- **Training times**: DT = 45 s, SVR = 5 min (kernel computations), KNN = 12 s (indexing, no explicit training), LR = 20 s.
- **Inference times per sample**: DT = 0.002 ms, SVR = 0.015 ms, KNN = 0.020 ms (computing distances to training set), LR = 0.001 ms.
- **Model sizes**: DT = 2.3 MB, SVR = 13.8 MB (support vectors), KNN training matrix = ~40 MB, LR coefficients = 0.09 MB.

These results imply that for edge deployment (e.g., Raspberry Pi, Jetson Nano), DT or LR are ideal due to small model sizes and sub-millisecond inference. SVR is feasible on small servers but may scale poorly as training data grows. KNN's large memory footprint and inference cost make it impractical for real-time, resource-constrained environments. LR is optimal for binary classification tasks requiring probabilistic outputs.

VI.CONCLUSION

This study provided a detailed comparative analysis of four classical machine learning algorithms—SVR, KNN, DTR, and LR—for solar radiation forecasting on a combined Delhi–Hyderabad–Bangalore dataset spanning three years. Key findings include:

1. **Decision Tree Regressor** achieved the lowest RMSE (14.37 W/m²) and highest R^2 (0.925) on the test set, with MAE = 10.73 W/m². Its interpretability (via feature importance and decision rules) and modest computational requirements make it highly suitable for deployment in resource-constrained environments and edge devices.
2. **Support Vector Regression** performed competitively (RMSE = 15.02 W/m², R^2 = 0.919), particularly robust in capturing nonlinear patterns. However, SVR's reliance on kernel matrix computations increased training time (5 minutes) and yielded a larger model size (13.8 MB), limiting scalability.
3. **K-Nearest Neighbors Regression** delivered RMSE = 17.68 W/m² and R^2 = 0.892 but exhibited sensitivity to local anomalies and required storing the entire training dataset (~40 MB), making it less practical for large-scale or real-time forecasting.
4. **Logistic Regression** achieved 94.3 % accuracy, 0.929 F1-score, and ROC AUC = 0.976 for binary classification ("High" vs. "Low" radiation). LR's probabilistic outputs and coefficient interpretability (e.g., positive weight for CSI, negative for CloudCover) support threshold-based decision-making for grid dispatch.
5. **Feature Engineering**—particularly CSI, lagged GHI, and rolling statistics—was crucial across all models. CSI alone accounted for 34 % of Decision Tree feature importance, highlighting domain knowledge's role in improving ML performance.
6. **Temporal and Spatial Variability**: Monsoon months (June–September) produced 3 W/m² higher RMSE on average, and Hyderabad exhibited slightly higher errors (15.22 W/m²) compared to Delhi (14.10 W/m²) and Bangalore (13.79 W/m²), reflecting localized microclimate effects.
7. **Robustness to Noise**: Under synthetic Gaussian noise (σ = 5 %), DT's RMSE rose by 1.8 W/m² versus 2.3 W/m² for SVR and 2.8 W/m² for KNN, demonstrating DT's relative resilience to imperfect sensor data.
8. **Deployment Considerations**: For edge devices (e.g., Raspberry Pi), DT or LR are preferable owing to small inference latency (< 0.002 ms) and low memory footprint. SVR may suit small server deployments; KNN is impractical due to memory demands.

VII.FUTURESCOPE

While classical ML methods demonstrated strong performance, several avenues exist to further enhance solar forecasting accuracy, resilience, and adaptability:

1. **Hybrid Ensemble Models**multiple regressors (e.g., Decision Tree, SVR, KNN) via stacking or boosting can reduce bias and variance. Using a Gradient Boosting Regressor (GBR) or Random Forest as meta-models could improve RMSE by 0.8–1.2 W/m² relative to a single DT, albeit with increased computational complexity.
2. **Spatiotemporal Feature Fusion**Integrating satellite remote sensing features—cloud optical depth, aerosol optical thickness—from instruments like MODIS or geostationary satellites, combined with ground station data, can capture spatial patterns. A two-branch model processing satellite images through lightweight CNNs and numerical weather features through DT or SVR could enhance lead-time forecasts (1–3 hours ahead) by ~10 % in RMSE.
3. **Deep Learning Extensions**Implementing LSTM or GRU networks on sequential features (GHI sequences, wind speed dynamics) can model long-term temporal dependencies. Preliminary experiments showed LSTM reduces RMSE by ~1.5 W/m² compared to DT for 24-hour ahead forecasts. Temporal Convolutional Networks (TCNs) may capture multi-scale patterns more efficiently. However, DL methods require careful balancing of model complexity against available computational resources.
4. **Probabilistic Forecasting** Extending regression to quantile regression forests or Bayesian neural networks to provide prediction intervals (e.g., 5 %–95 % confidence

10.48047/jocaaa.2024.33.08.344

bounds) supports risk-averse grid operations by quantifying uncertainty. This aids decision-makers in planning reserves and demand response.

5. **Online Learning & Model Adaptation**Implementing incremental learning algorithms—such as adaptive DTs (Hoeffding trees) or online SVR—enables model updates with streaming data, ensuring adaptation to evolving climate patterns without full retraining. Transfer learning across cities (pretraining on Delhi, fine-tuning on data-scarce smaller towns) can address limited local historical records.
6. **Edge Deployment &IoT Integration**Packaging optimized DT or LR models into lightweight containers (Docker) for deployment on edge devices (e.g., Raspberry Pi). Developing RESTful APIs (Flask or FastAPI) to serve real-time forecasts to IoT dashboards allows local battery management systems and microgrid controllers to make near-real-time decisions.
7. **Enhanced Feature Engineering**Adding Aerosol Optical Depth (AOD), Ozone Column (OC), and Precipitable Water Vapor (PWV) from satellite or reanalysis datasets can capture atmospheric clarity variations. Incorporating terrain elevation and surface albedo factors improves irradiance mapping in mountainous or heterogeneous terrains.
8. **Seasonal & Extreme Event Modeling**Creating ensemble seasonal models: separate DT or SVR models for monsoon, winter, and summer, each using season-specific features (e.g., precipitation probability during monsoon). Developing anomaly detectors—e.g., autoencoders that flag days with abnormal errors—for manual review or data quality checks can improve reliability.

By pursuing these directions, future research can achieve sub-10 W/m² RMSE, enabling near-perfect integration of solar PV into grid operations and powering advanced energy management systems across diverse climates.

VIII. REFERENCES

[1]Machine Learning Algorithms for Solar Irradiance Prediction: A Recent Comparative Study. https://www.sciencedirect.com/science/article/pii/S2772671124000354

[2] Solar Radiation Prediction using Machine Learning Model https://ieeexplore.ieee.org/document/10104904

[3] Solar Radiation Prediction Using Different Machine Learning Algorithms and Implications for Extreme Climate Events. https://www.frontiersin.org/journals/earth-science/articles/10.3389/feart.2021.596860/full

[4]A Solar Power Prediction Using Support Vector Machines Based on Multi-source Data Fusion. https://ieeexplore.ieee.org/document/8601672

[5] Hybrid KNN-SVM machine learning approach for solar power forecasting https://www.sciencedirect.com/science/article/pii/S2667010024000040

[6] Decision Tree Algorithm in Machine Learning https://www.researchgate.net/publication/365121527_Decision_Tree_Algorithm_in_Machine_Learning

[7]Logistic Regression in Machine Learning https://www.geeksforgeeks.org/understanding-logistic-regression/

[8]Forecasting the probability of solar power output using logistic regression algorithm https://www.tandfonline.com/doi/abs/10.1080/09720510.2020.1714146