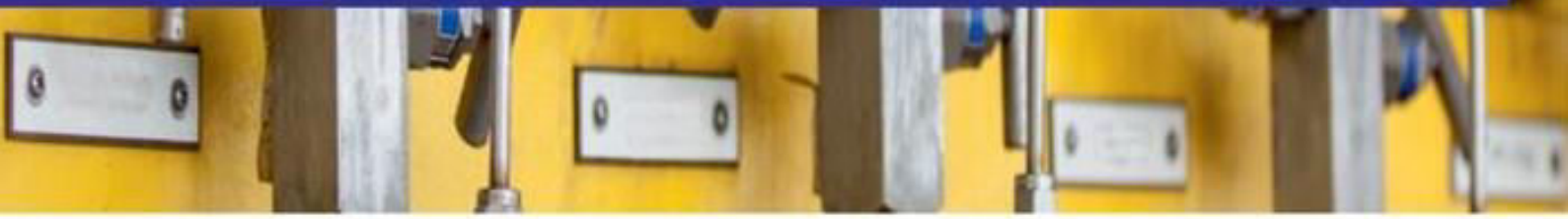


IJAREEIE

e-ISSN: 2278-8875
p-ISSN: 2320-3765

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

Volume 15, Issue 11, November 2026



INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 9.867

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (IJAREEIE)

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

PowerAI: A Gradient-Boosting Electricity Load Forecaster with an Interactive Dashboard for Appliance Impact Simulation and Energy-Saving Recommendation

Faiz Rangari, Swayam Jadhav, Sidhant Nelge, Dikshant Mane

Department of Electronics and Computer, Science Vidyalankar, Institute of Technology, Mumbai, India

Department of Electronics and Computer Science, Vidyalankar Institute of Technology, Mumbai, India

Department of Electronics and Computer Science, Vidyalankar Institute of Technology, Mumbai, India

Department of Electronics and Computer Science, Vidyalankar Institute of Technology, Mumbai, India

ABSTRACT: Household electricity forecasting is usually delivered as an error table rather than as a tool a consumer can use. This paper presents PowerAI, an interactive web dashboard that combines short-term load forecasting, appliance-level analysis, a what-if appliance impact simulator, historical data exploration and energy-saving recommendations in one interface, together with the forecasting back end that drives it. The forecasting engine uses four histogram-based gradient-boosting models trained on the UCI Individual Household Electric Power Consumption data (34,168 valid hourly observations, 2006–2010): a day-ahead model that uses only information at least 24 h old, an hour-ahead model, and 10% and 90% quantile models that give an 80% prediction interval. The models are served by a Flask REST API to an eight-module HTML, CSS and JavaScript front end built with Chart.js. On a chronologically held-out test period the day-ahead model attains an MAE of 0.415 kW, an RMSE of 0.575 kW and an RZ of 0.340, 18.5% lower MAE than a same-hour-yesterday baseline; the hour-ahead model reaches an MAE of 0.318 kW and an RZ of 0.572. The interval covers 82.5% of test hours. The appliance simulator, history module and saving tips are closed-form, synthetic or rule-based, and the weather inputs used in training are placeholders. The results show that a lightweight tree-based model gives a clear gain over naive and linear baselines on noisy single-household data, and that it can be exposed through a usable dashboard. They also show where the present prototype is not yet backed by measured data.

KEYWORDS: electricity load forecasting, gradient boosting, quantile prediction intervals, smart home, energy dashboard, appliance simulation, Flask, Chart.js.

I. INTRODUCTION

Residential electricity demand is rising with the adoption of air conditioning, electric water heating and electric-vehicle charging. For a household, the consequence is a monthly bill that is hard to anticipate and to attribute to individual devices; for a utility, it is a load curve with sharp evening peaks that must be served with costly peaking capacity. Short-term load forecasting addresses both problems, because a reliable prediction of the next hours or days lets consumers shift flexible loads and lets operators plan generation and procurement [1]–[3].

Despite a large body of research on forecasting models, results are rarely delivered to end users in a form that is interactive and easy to interpret. Many published systems stop at an error table, while commercial smart-meter portals typically show only past consumption. Statistical models are cheap but assume regular patterns, whereas deep networks learn richer patterns at a higher training and tuning cost [4]. For a single household with limited data, a tree-based model is a practical middle path.

This paper presents a project with two components:
1) A forecasting engine: four gradient-boosting models (day-ahead, hour-ahead and 10%/90% quantile) trained on real household data, with a leakage-safe feature set in which every consumption-derived feature is at least 24 h old for the day-ahead model.



| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

2) The PowerAI dashboard, an eight-module web application—Dashboard, Forecast, Analytics, Appliances, AI Model, Simulator, History and Save Energy—that displays the forecasts and model metrics, and adds a closed-form appliance impact simulator and a rule-based recommendation layer.

The models were trained on the first 85% of the hourly series and evaluated on the final 15%, which they never saw. The day-ahead model reduces MAE by 18.5% against a same-hour-yesterday baseline, and the hour-ahead model by 37.5%. The appliance, history and saving-tip modules are synthetic or rule-based, and this is stated throughout.

The rest of the paper is organised as follows. Section II reviews related work, Section III describes the system, Section IV gives the experimental setup, Section V reports the results, Section VI discusses them together with the limitations, and Section VII concludes.

II. RELATED WORK

Statistical forecasting. Autoregressive integrated moving-average (ARIMA) models and their seasonal variants remain a standard baseline for load forecasting because they are fast and interpretable [4]. They capture trend and periodicity but struggle with the nonlinear influence of weather and occupancy on household demand. Probabilistic forecasting, which reports a range as well as a point value, is increasingly recommended [2].

Gradient-boosted trees. Gradient boosting builds an additive ensemble of shallow regression trees by stagewise minimisation of a loss [5], [6]. Tree ensembles handle heterogeneous tabular features—hour, weekday, holiday flag, temperature—without extensive scaling and are frequently strong, inexpensive baselines. Histogram-based implementations, popularised by LightGBM [7] and available in scikit-learn [8], bin the features for speed and handle missing values natively. Replacing the squared error by the pinball loss turns a model into a conditional-quantile estimator [9], which gives prediction intervals without distributional assumptions.

Deep sequence models. Long short-term memory (LSTM) networks [10] model long-range temporal dependence through gated memory cells, and attention mechanisms [11] let the network weight the most informative past time-steps. Transformer architectures [12] replace recurrence with self-attention and have become competitive for multi-horizon forecasting. These models are not trained in the present work (Section VI).

Public datasets. Research on residential load forecasting commonly relies on the UCI Individual Household Electric Power Consumption data set [13], the GEFCom2014 competition data [14], the appliance-level UK-DALE data set [15] and the Pecan Street Dataport [16]. PowerAI trains on the UCI data; the other sources, and Indian grid reports [17], are listed in the application as candidates for future training.

III. SYSTEM DESIGN

A. Overview

PowerAI follows a four-layer design (Fig. 1). The presentation layer consists of a semantic HTML5 document (669 lines) and a CSS3 style sheet (818 lines) that implements a dark theme, a fixed sidebar and responsive card grids. The application-logic layer is a single JavaScript file (2,174 lines) containing the appliance manager, impact simulator, recommendation logic, history manager, synthetic data generators and an API client. The visualisation layer uses Chart.js 4.4.0 [18] on HTML5 canvas elements. The backend layer is a 203-line Flask application that loads the trained model bundle once at start-up, exposes JSON endpoints and also serves the front-end files, which avoids cross-origin problems. On page load the client requests /api/dashboard, /api/metrics and /api/health; a status badge shows “Live Model” when they succeed and “Demo Mode” (synthetic data) otherwise.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

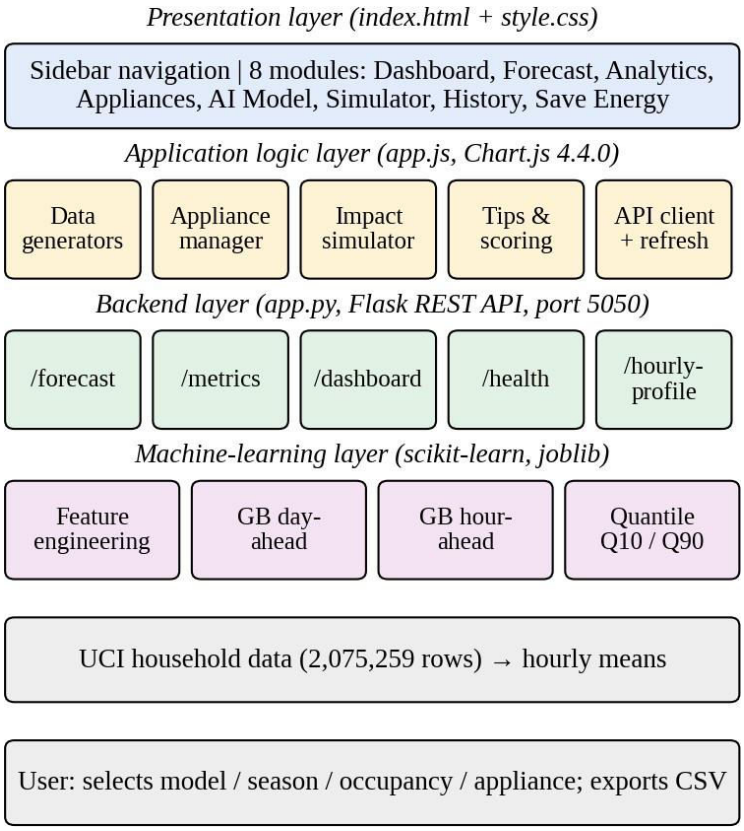


Fig. 1. Layered architecture of PowerAI. The front end calls a Flask API, which serves four gradient-boosting models trained on the UCI data.

B. Forecasting Engine

Let $y(\tau)$ be the hourly load at time τ. The day-ahead model estimates $y(\tau)$ using only observations at least 24 h old, so that a single model can forecast any of the next 24 hours from the last observed hour:

$$\hat{y}(\tau) = f(\, c(\tau),\, w(\tau),\, y(\tau-24),\, y(\tau-48),\, y(\tau-168),\, \bar{y}_7(\tau),\, m_{24}(\tau),\, m_{168}(\tau)\,) \quad (1)$$

$$\bar{y}_7(\tau) = (1/7)\, \Sigma_{i=1}^{7}\, y(\tau - 24i) \quad (2)$$

Here $c(\tau)$ are calendar features (hour, day of week, month, weekend and holiday flags, and sine and cosine of the hour), $w(\tau)$ are temperature, humidity and the 24-hour temperature change, and m_{24} and m_{168} are rolling means over 24 and 168 h of the series shifted by 24 h. This gives 16 features. The hour-ahead model adds $y(\tau-1)$, $y(\tau-2)$, $y(\tau-3)$ and their three-hour mean, giving 20 features. One feature function is shared by training and serving to prevent train/serve skew. All learners are scikit-learn HistGradientBoostingRegressor models [8]. The day-ahead and hour-ahead models minimise squared error. Two further day-ahead models are fitted with the pinball loss at quantile $q = 0.1$ and 0.9 [9]:

$$L_q(y, \hat{y}) = \max(\, q(y - \hat{y}),\, (q - 1)(y - \hat{y})\,) \quad (3)$$

Their outputs bound an 80% interval, clipped so that the lower bound never exceeds and the upper bound never falls below the point forecast. At request time the service appends 24 (or 1) future rows to the history, rebuilds the features and returns the forecasts; future weather is approximated by the value 24 h earlier.

C. Web Application

Navigation is handled without page reloads: a click on a sidebar item hides all section containers, reveals the selected one and updates the page title. Table I summarises the eight modules and states which are driven by the trained backend. When the backend is online, the main forecast chart, the monthly, weekday and seasonal charts, the model panel and the live 24-hour forecast are re-drawn from API data after the page has initialised with its synthetic defaults. Figs. 2 and 3 show the running application.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

TABLE I. FUNCTIONAL MODULES OF POWERAI

Module	Main functions and visual elements	Data source
Dashboard	KPI cards; actual-vs-forecast chart with interval band (last test week); load-distribution doughnut; weekly heat-map; anomaly chart; insight feed	Mixed
Forecast	Selectors for horizon (24 h–30 d), model, season and occupancy; live 24 h forecast with interval; MAPE badge; min/max and total indicators	Live model
Analytics	Monthly, day-of-week and seasonal consumption (UCI); temperature scatter (synthetic)	UCI + synthetic
Appliances	Eight toggleable appliance cards; hourly profile and energy-share charts	Synthetic
AI Model	Metrics panel, feature importance, training-loss curve, model comparison	Trained model
Simulator	What-if analysis for 13 appliance types: added kWh, bill, CO_2, before/after load	Closed-form
History	730 daily records with filters, search, pagination, four distribution charts and CSV export	Synthetic
Save Energy	Five reference datasets, energy score ring, savings chart and 12 saving tips	Rule-based

(a) Dashboard



(b) Forecast



(c) Analytics



(d) Appliances



Fig. 2. The PowerAI interface with the backend connected ("Live Model" badge): (a) Dashboard with KPI cards, forecast-vs-actual chart and load distribution; (b) Forecast with the live 24-hour gradient-boosting forecast and 80% interval; (c) Analytics with monthly, day-of-week and seasonal consumption from UCI data; (d) Appliances with toggleable device cards.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

(e) AI Model



(f) Simulator



(g) History



(h) Save Energy

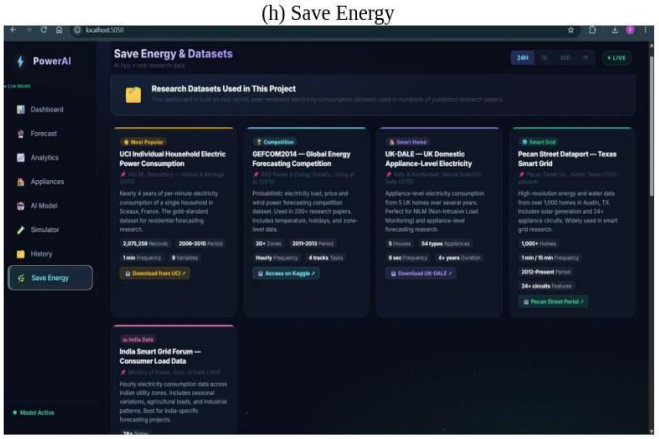


Fig. 3. The PowerAI interface (continued): (e) AI Model with test metrics, feature importance, loss curve and model comparison; (f) Simulator for a 1.5-ton air conditioner used 4 h/day; (g) History with synthetic 2023–2024 records; (h) Save Energy & Datasets.

D. Appliance Impact Simulator

The simulator answers the question "what will a new appliance do to my bill?". The user chooses one of 12 preset devices or a custom power rating P in watts, the daily usage h in hours, the number of days per month d and the tariff τ in ₹/kWh. The added monthly energy, cost and emissions are

$$E_{add} = (P / 1000) \cdot h \cdot d \quad (4)$$

$$\Delta C = E_{add} \cdot \tau, \quad C_{new} = E_{base} \cdot \tau + \Delta C \quad (5)$$

$$MCO2 = E_{add} \cdot \varepsilon \quad (6)$$

where E_{base} = 1,281 kWh per month is the household baseline (₹ 8,967 at ₹ 7/kWh) and ε = 0.82 kg CO_2/kWh is the emission factor assumed for the Indian grid. For the hourly comparison, the appliance load is placed in a window starting at 08:00 and lasting h hours (capped at 23:00), or held constant if h ≥ 24, and is added to a 24-value base profile b(h).

E. Recommendation and Scoring Layer

The insight feed displays six context-style messages, such as pre-cooling before an evening peak or shifting dishwasher and laundry loads after 11 PM. The Save Energy page adds an energy-efficiency score (68/100, Grade B) and a bar chart of indicative monthly savings: solar with net metering (₹ 650), HVAC optimisation (₹ 380), star-rated appliances (₹ 210), schedule shifting (₹ 120), insulation and sealing (₹ 95) and LED lighting (₹ 85), a total of ₹ 1,540. Twelve tip cards give a category, an estimated saving and an effort label; for example, raising the air-conditioner set-point by 1 °C is stated to save about 6% of cooling energy. These are static rules and estimates entered in the interface, not outputs of a learned recommender.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

IV. EXPERIMENTAL SETUP

A. Datasets

Real load data. The forecasting models and the analytics views use the UCI Individual Household Electric Power Consumption data set [13]: 2,075,259 one-minute measurements from one French household between 16 December 2006 and 26 November 2010. The target is Global$_{active_power}$ (kW). Missing values (coded "?") are parsed as NaN and the series is resampled to hourly means; because kW averaged over one hour equals kWh, the hourly target is treated as energy per hour. After resampling, 34,168 hours contain valid data (98.8% completeness) and 1,417 days are complete. Table II summarises the data and the split.

TABLE II. DATA USED TO TRAIN AND EVALUATE THE FORECASTER

Item	Value
Source	UCI household power [13]
Raw records (1 min)	2,075,259
Valid hourly observations	34,168 (98.8%)
Period	16 Dec 2006 – 26 Nov 2010
Complete days	1,417
Mean daily energy	26.2 kWh
Mean daily peak	2.86 kW
Training block ends	9 Nov 2009 14:00
Validation block ends	13 Apr 2010 09:00
Test block	13 Apr 2010 10:00 – 26 Nov 2010 21:00
Split (chronological)	75% / 10% / 15%

Synthetic data. The modules not backed by the trained model derive their series from a 24-value base profile b(h) in kW with a night minimum of 0.6 kW (02:00–03:00), a morning shoulder of 2.2 kW (08:00) and an evening maximum of 3.5 kW (19:00), summing to 41.4 kWh per day. A synthetic observation at hour h is

$$x(h) = max(0.3,\ b(h) + \varepsilon + w) \quad (7)$$

where ε is uniform noise in [−0.2, 0.2] kW and w is a weekend uplift drawn from [0, 0.3] kW. The History module generates 730 daily records from 1 January 2023, scaling the profile by a season multiplier (summer 1.25, winter 1.10, monsoon 0.90, spring 0.80), a weekend factor of 1.15 and, on 13 pre-selected days, an anomaly factor in [1.35, 1.75]. The Appliances module models eight devices by rated power P (kW) and daily usage t (h), with daily energy E = P·t; the six devices active by default draw 25.6 kWh per day, of which the air conditioner contributes 14.4 kWh (56%).

B. Training Details

The models were implemented with scikit-learn and trained on a CPU. Rows with a missing target or missing key lags are removed and the remainder is split chronologically (Table II). Because UCI contains no weather, training used placeholder values: a French monthly-mean temperature profile plus Gaussian noise, and a constant 65% humidity. A grid over learning rate {0.05, 0.1} and maximum depth {4, 6, unlimited} was scored on validation MAE; the selected configuration is listed in Table III. Final models are refit on training plus validation data. Feature importance is measured by permutation on the test set (three repeats, MAE scoring), clipped at zero and normalised.

TABLE III. FORECASTER TRAINING HYPER-PARAMETERS

Setting	Value
Learner	HistGradientBoostingRegressor
Learning rate	0.1 (grid: 0.05, 0.1)
Maximum depth	Unlimited (grid: 4, 6, none)
Iterations	300
L2 regularisation	1.0
Minimum samples per leaf	20



| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

Quantile models	q = 0.1 and q = 0.9
Ridge baseline	Median impute, scale, α = 1
Model selection	Lowest validation MAE
Final fit	Training + validation

C. Evaluation Metrics

Baselines are same hour yesterday, same hour last week, the seven-day same-hour average, and ridge regression. Model quality is reported with standard metrics [19], where y_t is the observed and $\hat{y}_t$ the predicted load over n test points and $\bar{y}$ is the mean observation. MAPE is computed only over hours above 0.05 kW to avoid division by near-zero values. Interval quality is the empirical coverage of the nominal 80% interval.

$$MAE = (1/n)\,\Sigma_t\,|y_t - \hat{y}_t| \quad (8)$$

$$RMSE = \sqrt{[\,(1/n)\,\Sigma_t\,(y_t - \hat{y}_t)^2\,]} \quad (9)$$

$$MAPE = (100\%/n)\,\Sigma_t\,|(y_t - \hat{y}_t)/y_t| \quad (10)$$

$$R^2 = 1 - \Sigma_t(y_t - \hat{y}_t)^2 / \Sigma_t(y_t - \bar{y})^2 \quad (11)$$

The simulator and the recommendation layer are closed-form and are assessed by worked examples rather than by error metrics.

V. RESULTS

A. Forecast Accuracy

Table IV and Fig. 4(c) summarise the test-set performance. The day-ahead gradient-boosting model achieves an MAE of 0.415 kW, a 5.4% reduction relative to ridge regression, 3.7% relative to the strongest naive baseline (the seven-day same-hour average) and 18.5% relative to same-hour-yesterday. Its RMSE of 0.575 is the lowest among day-ahead methods. Adding the three most recent hours reduces MAE to 0.318 kW (37.5% below yesterday) and raises RZ from 0.340 to 0.572. High MAPE values arise because household load often falls to very low levels, so MAE and RMSE are the more informative measures.

TABLE IV. TEST-SET METRICS ON THE UNSEEN PERIOD

Model	MAE	RMSE	MAPE %	R^2
Same hour yesterday	0.509	0.756	66.67	−0.141
Same hour last week	0.551	0.796	76.78	−0.264
7-day same-hour avg.	0.431	0.601	63.08	0.280
Ridge (day-ahead)	0.439	0.591	65.97	0.303
Grad. boost. (day)	0.415	0.575	61.20	0.340
Grad. boost. (hour)	0.318	0.463	42.81	0.572

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

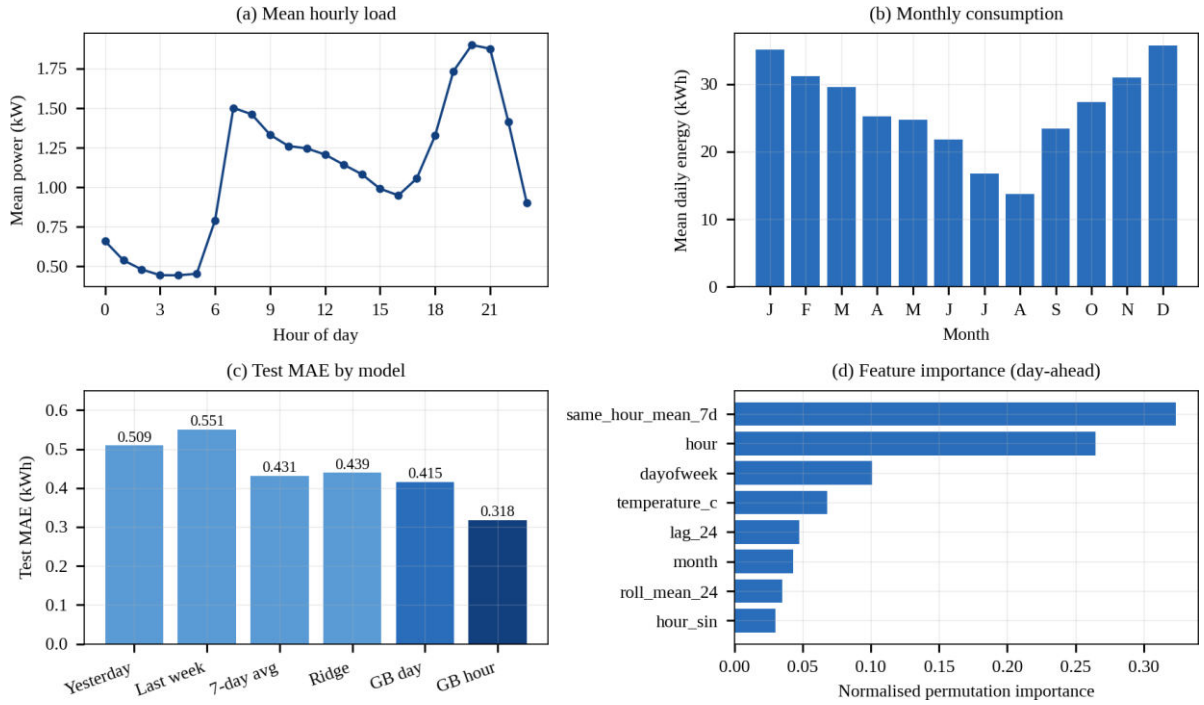


Fig. 4. Data and model results: (a) mean hourly load over the whole data set; (b) mean daily energy by calendar month; (c) test MAE of baselines and gradient-boosting models; (d) permutation importance of the eight most important day-ahead features.

B. Interval, Feature Importance and Learning Curve

The 10%–90% interval covers 82.5% of test hours against a nominal 80%, i.e. close to calibrated and slightly conservative. Fig. 5(a) shows that the forecast tracks the daily cycle but underestimates sharp isolated peaks. Permutation importance (Fig. 4(d)) is led by the seven-day same-hour mean (0.324) and hour of day (0.265), followed by day of week (0.101) and temperature (0.068); the weekend flag, holiday flag and humidity score zero. Validation loss is lowest (0.473) near iteration 48 and rises slightly to 0.484, while training loss keeps falling to 0.323 (Fig. 5(b)), so the model overfits mildly and early stopping may help.

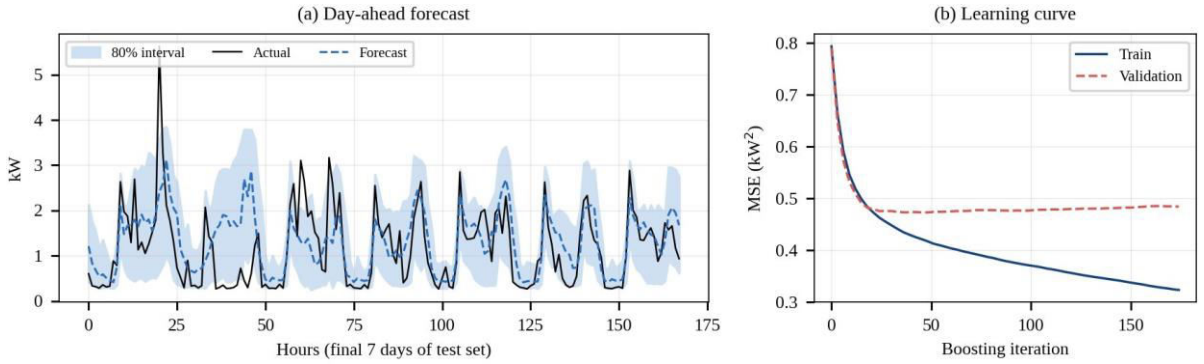


Fig. 5. (a) Actual versus day-ahead forecast with the 80% interval over the final 168 test hours; (b) training and validation MSE versus boosting iteration.

C. Consumption Characteristics

The real data show a low overnight load of about 0.44 kW, a morning peak of 1.50 kW at 07:00 and a larger evening peak of 1.90 kW at 20:00 (Fig. 4(a)). Mean daily energy ranges from 13.7 kWh in August to 35.7 kWh in December (Fig. 4(b)), consistent with winter heating demand and summer absence. Saturday and Sunday average 29.9 and 29.3 kWh against about 24.8 kWh on weekdays, roughly 19% higher.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

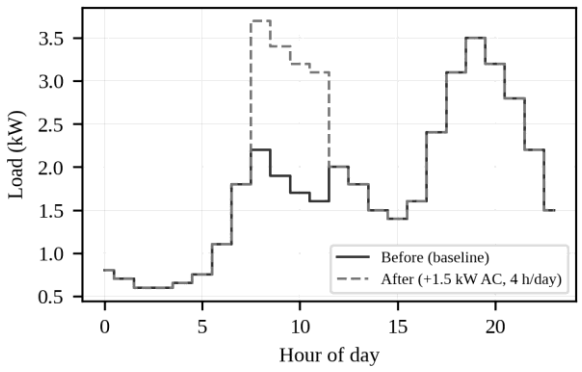
D. Appliance Simulator

Table V gives worked simulator results for five devices at ₹ 7/kWh over 30 days, and Fig. 6 shows the before/after hourly load for the air-conditioner case. The EV charger is the heaviest addition at 396 kWh and ₹ 2,772 per month, whereas a ceiling fan used 8 h/day adds only ₹ 126.

TABLE V. SIMULATOR EXAMPLES (30 DAYS, ₹7/kWh)

Appliance	P (W)	h/day	ΔE (kWh)	ΔC (₹)	CO_2 (kg)
AC (1.5 ton)	1500	4	180.0	1,260	147.6
EV charger	3300	4	396.0	2,772	324.7
Water heater	2000	1	60.0	420	49.2
Refrigerator	150	24	108.0	756	88.6
Ceiling fan	75	8	18.0	126	14.8

Fig. 6. Before/after hourly load for a 1.5-ton air conditioner used 4 h/day (simulator output).



VI. DISCUSSION

A. Why Gradient Boosting

The interface originally listed LSTM, ARIMA, XGBoost and Transformer models with illustrative error values. The present system replaces those values by measured results from one trained family. Gradient boosting was chosen because it works with modest amounts of tabular data, trains in minutes on a CPU, needs no feature scaling, handles missing weather values natively, and is easy to explain through feature importance. The seven-day same-hour mean and hour of day dominate, which shows that the model mainly learns the household's weekly routine; the 3.7% gain over the seven-day average baseline shows how much of the day-ahead signal that routine already carries.

B. Real and Synthetic Parts

Table VI contrasts the trained forecasting engine with the prototype modules. The forecasting engine is causal in the sense that every day-ahead input is available 24 h in advance and its results are measured on held-out data. The prototype modules illustrate interface and calculation logic, but their numbers are not measurements.

TABLE VI. CHARACTERISTICS OF THE TWO PARTS OF POWERAI

	Forecasting engine	Prototype modules
Approach	Gradient boosting	Closed-form / rules
Data	UCI, 34,168 h	Synthetic profile
Parameters	4 GB models + ridge	None
Weather	Placeholder	Not used
Levels	Day / hour / interval	User selection
Evaluation	Held-out metrics	Worked examples

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

C. Limitations

Several limitations qualify these results. (i) The UCI data contain no weather: training used a French monthly-mean temperature plus unseeded Gaussian noise and a constant humidity, so the temperature feature acts largely as a noisy month proxy. (ii) The serving code supplies NaN weather for the UCI data, which differs from the placeholder values seen in training; its effect on live forecasts has not been verified. (iii) Season labels follow an Indian convention but are applied to French data, so seasonal aggregates should be read by calendar month; the holiday list covers Indian dates for 2023–2024 and has no effect on 2006–2010. (iv) The test block spans April–November and excludes the highest-demand winter months, and results are averaged over the whole block. (v) The data describe one household and may not generalise. (vi) The dashboard's "next 24 hours" forecast starts from the last observation (26 November 2010), so it is a demonstration, not a real-time forecast. (vii) Some interface elements still reflect the prototype: the AI Model page shows a static LSTM-attention architecture panel beside metrics from the gradient-boosting model, the Forecast page's model selector still reads "LSTM Neural Network" while the chart header reports gradient boosting, the "Model accuracy" tile shows RZ × 100 (34.0%) rather than a classification accuracy, and the temperature scatter on the Analytics page is synthetic. (viii) The History page is headed "Past data used to train the AI model" and reports 5,840 records for 2023–2024, but these are synthetic records generated in the browser; the models were trained on the UCI data. (ix) The appliance, simulator, insight and saving modules use random generators, fixed constants and static estimates. (x) The tariff is a flat rate, the emission factor is a constant, and saving estimates were not validated against measurements.

VII. CONCLUSION AND FUTURE WORK

This paper presented PowerAI, a web dashboard for household electricity forecasting, appliance impact simulation and energy-saving guidance, together with a gradient-boosting forecasting engine trained on real data. On the held-out test period, the day-ahead model reduces MAE by 18.5% against same-hour-yesterday and the hour-ahead model by 37.5%, and the quantile models give an 80% interval that covers 82.5% of hours. The dashboard exposes these results alongside synthetic and rule-based modules whose limits are stated explicitly. Future work will (1) replace placeholder weather with measured and forecast weather; (2) train and compare LSTM-attention and Transformer models on the UCI and UK-DALE data sets using rolling-origin evaluation across all seasons; (3) add early stopping and a wider hyper-parameter search; (4) ingest live smart-meter readings and replace the synthetic appliance and history modules with measured data; (5) support time-of-use tariffs and region-specific emission factors; and (6) evaluate usability and energy-saving effect in a user study.

REFERENCES

[1] R. J. Hyndman and G. Athanasopoulos, Forecasting: Principles and Practice, 2nd ed. Melbourne, Australia: OTexts, 2018.
[2] T. Hong and S. Fan, "Probabilistic electric load forecasting: A tutorial review," Int. J. Forecasting, vol. 32, no. 3, pp. 914–938, 2016.
[3] T. Hong, P. Pinson, and S. Fan, "Global Energy Forecasting Competition 2012," Int. J. Forecasting, vol. 30, no. 2, pp. 357–363, 2014.
[4] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, Time Series Analysis: Forecasting and Control, 5th ed. Hoboken, NJ, USA: Wiley, 2015.
[5] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, San Francisco, CA, USA, 2016, pp. 785–794.
[6] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," Ann. Statist., vol. 29, no. 5, pp. 1189–1232, 2001.
[7] G. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017, pp. 3146–3154.
[8] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.
[9] R. Koenker and G. Bassett, "Regression quantiles," Econometrica, vol. 46, no. 1, pp. 33–50, 1978.
[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
[11] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in Proc. Int. Conf. Learning Representations (ICLR), San Diego, CA, USA, 2015.
[12] A. Vaswani et al., "Attention is all you need," in Advances in Neural Information Processing Systems, vol. 30, 2017.



| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 11, November 2026 ||

| DOI:10.15662/IJAREEIE.2026.1511011 |

[13] G. Hébrail and A. Bérard, "Individual household electric power consumption," UCI Machine Learning Repository, 2012. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption

[14] T. Hong, P. Pinson, S. Fan, H. Zareipour, A. Troccoli, and R. J. Hyndman, "Probabilistic energy forecasting: Global Energy Forecasting Competition 2014 and beyond," Int. J. Forecasting, vol. 32, no. 3, pp. 896–913, 2016.

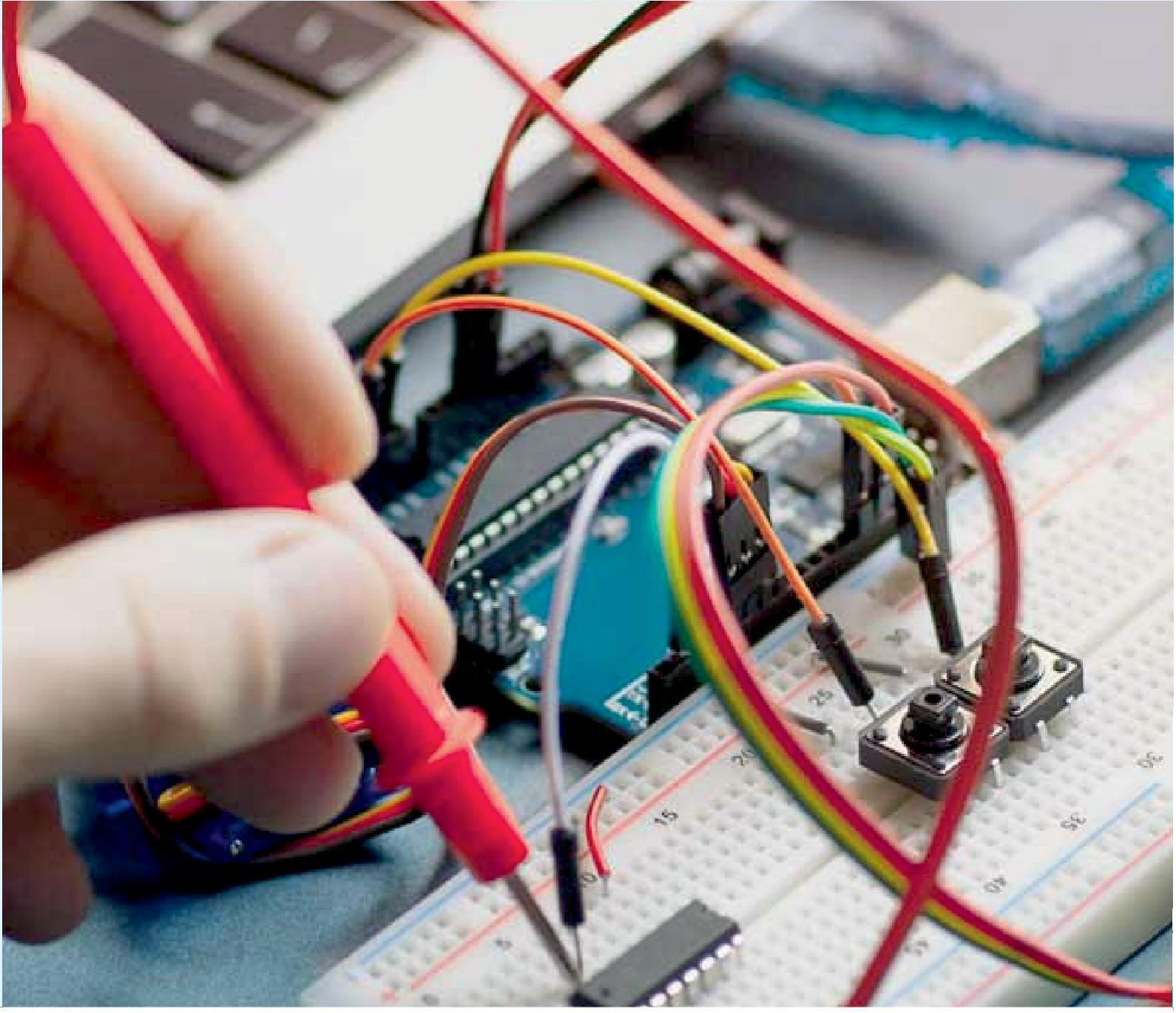
[15] J. Kelly and W. Knottenbelt, "The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes," Scientific Data, vol. 2, Art. no. 150007, 2015.

[16] Pecan Street Inc., "Dataport." [Online]. Available: https://dataport.pecanstreet.org/

[17] Power System Operation Corporation Ltd. (POSOCO), "Daily reports." [Online]. Available: https://www.posoco.in/reports/daily-reports/

[18] Chart.js Contributors, "Chart.js documentation, v4.4.0." [Online]. Available: https://www.chartjs.org/docs/

[19] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," Int. J. Forecasting, vol. 22, no. 4, pp. 679–688, 2006.



INNO SPACE
SJIF Scientific Journal Impact Factor

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

International Journal of Advanced Research

in Electrical, Electronics and Instrumentation Engineering

9940 572 462 6381 907 438 ijareeie@gmail.com

Scan to save the contact details