**PowerPulse: Household Energy Usage Forecast**

**📍 Objective**

To design a machine learning model that accurately predicts household energy usage using historical data, enabling users to optimize consumption and energy providers to plan more efficiently. The solution aims to support demand forecasting, anomaly detection, and environmental sustainability.

**🏠 Domain: Energy and Utilities**

**🎯 Business Use Cases**

* Household Energy Management: Inform users about consumption patterns to encourage bill savings and sustainable habits.
* Provider Demand Forecasting: Anticipate regional energy demands for efficient grid distribution and pricing.
* Anomaly Detection: Flag abnormal usage spikes signaling possible leaks, device failures, or unauthorized usage.
* Smart Grid Integration: Feed insights into smart systems for real-time optimization.
* Environmental Impact: Minimize wastage and contribute to carbon footprint reduction.

**🔍 Approach & Methodology**

**1. 📊 Data Understanding & EDA:**

* Load the Individual Household Electric Power Consumption dataset.
* Analyze missing values, distributions, seasonality, and correlations.
* Visualize hourly/daily consumption patterns with line plots, histograms, and heatmaps.

**2. 🧹 Data Preprocessing:**

* Missing Values: Impute or drop as needed.
* Date Parsing: Extract hour, day, month, weekday, and is\_weekend.
* Feature Scaling: Use MinMaxScaler or StandardScaler for uniformity.
* Time-Series Parsing: Convert to datetime index for rolling calculations.

**3. 🏗️ Feature Engineering:**

* Rolling Statistics: Moving averages (e.g., 3-hour, 24-hour).
* Peak Hour Flags: Tag hours of elevated consumption.
* Lag Features: For autoregressive modeling.
* Weather Integration (if available): Add temperature, humidity as regressors.

**4. 🤖 Model Development:**

* Train/Test Split: Use chronological split to prevent data leakage.
* Models Used:
* Linear Regression (baseline)
* Random Forest Regressor
* XGBoost or LightGBM
* Neural Network (Keras or PyTorch)
* Hyperparameter Tuning: Use GridSearchCV or Optuna for fine-tuning.

**5. 📈 Evaluation Metrics:**

| Metric | Purpose | | RMSE | Penalizes large errors | | MAE | Measures average absolute deviation | | R² Score | Indicates explained variance | | Feature Importance | Highlights influencing factors |

**📊 Key Visualizations:**

* Power Consumption vs Time → Line plot (daily/hourly trends)
* Seasonal Pattern Detection → Monthly/weekly box plots
* Feature Importance → Bar chart using model’s output
* Prediction vs Actuals → Overlayed time-series graphs

**🗂️ Deliverables:**

* Source Code:
* Python Notebook (Jupyter or .py) with comments.
* GitHub Repository organized with proper version control.
* Report Document:
* Methodology summary
* Key findings and insights
* Model comparison and rationale
* Visualizations:
* Embedded within the report or exported as images for dashboard use.

**Conclusion:**

Here the best performing model is XGBoosting regression model which has r2 score with 0.999 and lowest MAE and RMSE score (0.01 and 0.02 respectively) amongst linear regression, random forest regression, gradient boosting regression and neural networks regression.