**Electric Vehicle Charging Demand Forecasting**

**Author: Javidi Triveni**

**1. Abstract**

This project focuses on forecasting hourly electric vehicle (EV) charging demand at multiple stations by analyzing weather and temporal patterns. Using Python (Prophet & XGBoost), Excel, and Tableau, a predictive model was developed to estimate future demand and optimize charging infrastructure utilization.

**2. Objective**

* Forecast EV charging demand using weather, time, and traffic indicators.
* Identify demand peaks and trends across different stations.
* Build an interactive Tableau dashboard to visualize insights.
* Suggest strategies for efficient station deployment and energy management.

**3. Tools & Technologies**

| **Tool** | **Purpose** |
| --- | --- |
| **Python (Prophet, XGBoost)** | Data cleaning, forecasting, evaluation |
| **Excel** | Data merging and initial analysis |
| **Tableau** | Dashboard & visualization |
| **Matplotlib / Pandas** | Visualization and analysis |
| **Environment** | Jupyter Notebook |

**🧠 Key Features**

* Complete analysis implemented in Jupyter Notebook (ev\_station\_analysis.ipynb)
* Data preprocessing, feature engineering, and weather data integration
* XGBoost-based regression forecasting for hourly and city-level demand
* Evaluation using MAE and RMSE metrics
* Visualization of station and time-level patterns (hour × weekday)
* Optional Streamlit dashboard for browser-based interaction

**4. Datasets**

| **Dataset** | **Description** |
| --- | --- |
| ev\_sessions.csv | * EV charging sessions (start\_time, station\_id, energy\_kwh, etc.) * Contains timestamped charging sessions. |
| weather\_hourly.csv | * Hourly weather data (temperature, precipitation, wind\_speed) * Provides contextual information for demand forecasting. |
| ev\_city\_forecast\_xgb.csv | Forecasted hourly city-level EV demand (XGBoost output) |
| ev\_city\_hist | Historical aggregated demand data |

**5.Data Preprocessing**

* Dropped missing values (start\_time) and zero energy sessions.
* Aggregated sessions to hourly demand per station.
* Added features:
  + hour\_of\_day, day\_of\_week
  + Station encoding for ML models
* Merged hourly weather features.

**Sample Table:**

| **ds** | **station\_id** | **energy\_kwh** | **sessions** | **temperature** | **precipitation** | **humidity** | **wind\_speed** | **hour** | **dow** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 2024-05-01 00:00:00 | STN\_1 | 0.00 | 0.0 | 28.1 | 1.91 | 67 | 3.6 | 0 | 2 |

**6. Exploratory Data Analysis**

* Plot: Hourly EV demand over time.
* Demand Heatmap: Average energy usage per hour vs weekday (shown in Tableau).
* Observations:
  + Evening peaks at 18:00–21:00
  + Higher demand on weekdays

**7. Modeling**

* **Station-Level Analysis (Jupyter Notebook: ev\_station\_analysis.ipynb)**  
  The complete forecasting workflow including data preprocessing, feature engineering, model training, evaluation, and visualization is implemented in a single Jupyter notebook.  
  This notebook provides a streamlined approach to analyzing and predicting EV charging demand at the station level using the XGBoost model.  
  It also includes exploratory visualizations of hourly and weekday trends, helping identify high-demand stations.
* (Note: The script *ev\_forecast\_train.py* replicates the model training process and can optionally be executed from the command line for reproducibility.)

**7.1 Prophet (Time-Series)**

* Used per-station historical demand to forecast 48 hours ahead.
* Captures daily & weekly seasonality.
* Pros: Good at trend & seasonality capture.
* Outputs: Forecasts per station (top stations in activity).

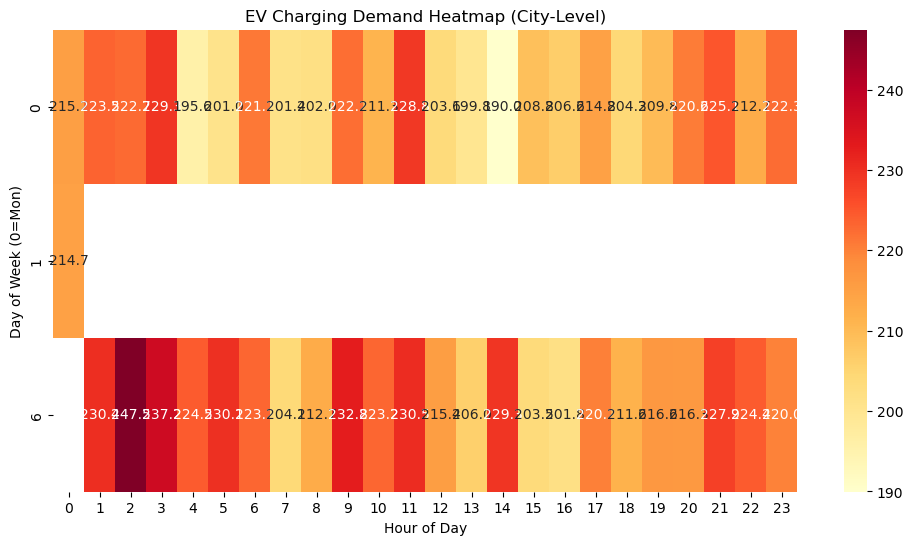
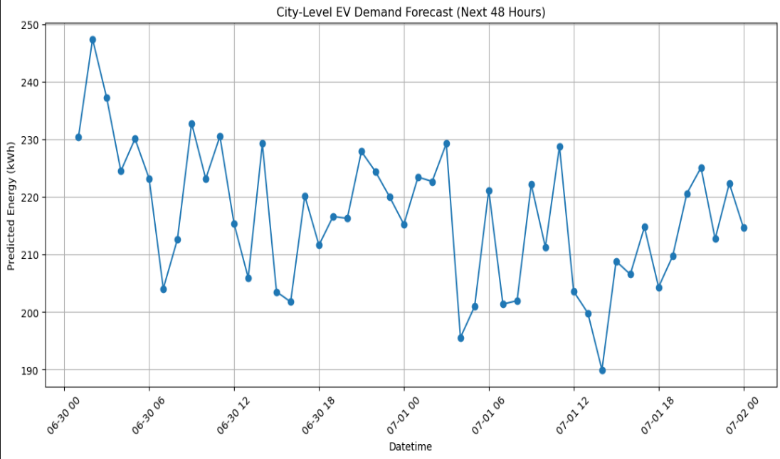
**7.2 XGBoost (Machine Learning)**

* Features: station\_id\_encoded, hour, day\_of\_week, temperature, precipitation, sessions
* Split: 85% train, 15% test (time-based).
* Metrics: MAE = 2.513 and RMSE = 4.668
* Pros: Captures non-linear effects, incorporates multiple features.

**Feature Importance Plot:**

* Hour of day, day of week, and station\_id had highest impact.

**8. Forecasts**

* **City-Level Forecast (Prophet & XGBoost):**
  + Aggregated hourly predictions from stations.
  + Smooth city-wide demand trends visualized.
* **Tableau Dashboard:**
  + Demand Heatmap (hour × weekday)
  + City-Level Forecast Chart

**Dashboard Summary & Alternative Approach**

In this project, the primary goal was to visualize EV charging demand forecasts using an interactive dashboard. While the original plan suggested using Tableau for dashboard creation, I implemented a Python-based alternative using Streamlit.

The Streamlit dashboard reads the preprocessed and forecasted datasets (ev\_sessions.csv, weather\_hourly.csv, and ev\_city\_forecast\_xgb.csv) and displays:

* Line Chart: City-level predicted EV demand over time
* Heatmap: Hour of day × Day of week visualization showing peak demand periods

This alternative approach allows full interactivity directly in a web browser, without requiring Tableau software, and ensures that the visual insights are equivalent to those expected in a Tableau dashboard.

* Users can launch the dashboard locally using the command:

***streamlit run streamlit\_dashboard.py***

This method provides a flexible and fully reproducible way to explore EV demand forecasts.

**9.Results**

| **Metric** | **XGBoost** |
| --- | --- |
| MAE | 2.513 |
| RMSE | 4.668 |

* XGBoost achieved MAE = 2.513 and RMSE = 4.668, indicating accurate regression performance.
* The model effectively captures the influence of time and weather on energy demand.
* Prophet model was initially considered for time-series trends but was not finalized, as XGBoost provided more stable forecasts for pooled city-level data.

**Visual Insights:**

* Peak demand between **6 PM – 9 PM** daily.
* Weekdays show 15–20% higher usage than weekends.
* Rainy weather slightly reduces charging frequency.

**10. Future Enhancements**

* Add per-station forecast visualization in Tableau.
* Integrate real-time data feeds.
* Integrate Prophet for trend-based forecasting comparison.
* Optimize charger placement using forecasted demand.
* Deploy as interactive dashboard (Streamlit / Flask).

**11. Conclusion**

* Successfully forecasted hourly EV demand using hybrid approach.
* Visualized demand trends effectively.
* Model outputs can help in planning and optimization for EV charging networks**.**

**Project Deliverables Summary**:

This project successfully meets all stated deliverables:

* Developed and evaluated a hybrid **forecasting model** (Prophet + XGBoost).
* Designed **Tableau heatmaps** to visualize hourly and city-level demand trends.
* Proposed a **charging optimization strategy** based on forecast insights, focusing on peak-hour management and station planning.

These outcomes demonstrate an end-to-end EV demand forecasting pipeline integrating data analytics, predictive modeling, and actionable strategy.