Microsoft Elevate

CAPSTONE PROJECT

GREEN AI ENERGY INTELLIGENCE

PRESENTED BY

STUDENT NAME: SUVASINI

COLLEGE NAME: CMR UNIVERSITY

DEPARTMENT: CSE (COMPUTER SCIENCE & ENGINEERING)

EMAIL ID: SUVASINIHULEPPA95@GMAIL.COM

OUTLINE:

- **Problem Statement** (Should not include solution)
- **Proposed System/Solution**
- **System Development Approach** (Technology Used)
- **Algorithm & Deployment**
- **Result (Output Image)**
- **Conclusion**
- **Future Scope**
- **References**

PROBLEM STATEMENT:

- Rising energy consumption in households and industries leads to increased electricity costs and higher carbon emissions, contributing to climate change.
- Consumers lack real-time visibility into their energy usage patterns and have no intelligent tool to predict future consumption or suggest optimization strategies.
- Traditional energy monitoring systems only display historical usage data without providing actionable insights or AI-driven recommendations for reduction.
- There is no accessible, data-driven platform that combines energy prediction, carbon footprint calculation, cost estimation, and optimization recommendations in a single solution.
- Without predictive analytics, consumers cannot identify peak usage hours, seasonal trends, or inefficient appliance patterns — leading to unnecessary energy waste.

PROPOSED SOLUTION:

- The proposed system is a **Green AI Energy Intelligence Platform** — a machine learning-based web application that predicts energy consumption, calculates carbon emissions, and provides optimization recommendations. The solution consists of:
- **Energy Prediction Engine:**
- Uses a Random Forest Regressor trained on real-world household power consumption data to predict energy usage based on time-based and electrical features.
- **Carbon Footprint Calculator:**
- Converts predicted energy consumption into CO_2 emissions using a standard emission factor (0.82 kg CO_2 per kWh) to quantify environmental impact.
- **Optimization Engine:**
- Provides intelligent recommendations to reduce energy usage (e.g., reducing AC hours, switching to LED, avoiding standby power) based on consumption levels.
- **Interactive Dashboard:**
- A Streamlit-based web interface with input sliders for parameters (Hour, Day, Month, Reactive Power, Electricity Cost) and real-time AI predictions with visualizations.

SYSTEM DEVELOPMENT APPROACH (TECHNOLOGY USED):

- **Programming Language:**

Python 3.12 — used for data processing, model training, visualization, and web application development.

- **ML Libraries:**

Scikit-learn (Random Forest, Linear Regression, KMeans Clustering), Pandas, NumPy for data manipulation and model training.

- **Visualization:**

Matplotlib for generating charts (scatter plots, line graphs, bar charts, heatmaps, histograms) and SHAP for model explainability.

- **Web Framework:**

Streamlit — used to build the interactive dashboard with real-time input parameters, cost projections, and environmental impact metrics.

- **Model Persistence:**

Joblib — used to serialize and save the trained Random Forest model (energy_model.pkl) for deployment.

- **Development Environment:**

Google Colab for model training and experimentation; deployed on a cloud platform with Streamlit for user-facing application

DATASET & PREPROCESSING:

- **Dataset:**

UCI Household Electric Power Consumption Dataset — contains 2,075,259 minute-level measurements of household energy usage collected over 4 years.

- **Features Used:**

Hour of Day, Day of Month, Month, Global Reactive Power — extracted via feature engineering from raw date/time and electrical readings.

- **Target Variable:**

Global Active Power (kW) — represents the total active energy consumed by the household per minute.

- **Data Cleaning:**

Missing values (marked as '?') were removed; data types were converted to float; datetime features were parsed and extracted for temporal analysis.

- **Train-Test Split:**

80/20 split using scikit-learn's train_test_split for model training and evaluation.

ALGORITHM & DEPLOYMENT:

- **Model 1 — Linear Regression:**

Baseline model trained on time-based features. Achieved MAE of 0.794 with fast training time (0.31 seconds).

- **Model 2 — Random Forest Regressor (Selected):**

Ensemble model with 50 estimators. Achieved significantly better MAE of 0.329 — selected as the production model due to superior accuracy.

- **KMeans Clustering:**

Applied unsupervised clustering (K=3) to categorize energy usage into Low, Medium, and High consumption patterns for user classification.

- **SHAP Explainability:**

SHAP (SHapley Additive exPlanations) values computed to identify which features contribute most to energy predictions — improving model transparency.

- **Optimization Simulation:**

A 15% energy reduction scenario was simulated to project cost savings, carbon reduction, and cumulative energy savings over 12 months.

RESULT:

The Green AI Energy Intelligence Platform was successfully built and deployed. Key outcomes:

- **Random Forest model** achieved MAE of 0.329 — significantly outperforming Linear Regression (MAE 0.794) on energy consumption prediction.
- **Energy Metrics Dashboard** displays predicted energy (kWh), estimated cost (₹), and carbon emission (kg CO_2) in real-time based on user input parameters.
- **Optimization Impact** shows energy saved (kWh), cost saved (₹), and carbon reduced (kg CO_2) — projecting a 15% reduction through AI-driven recommendations.
- **12-Month Projections** with interactive charts show optimized vs current energy and cost trends, plus cumulative savings visualization.
- **Carbon & Environmental Impact** section calculates 1-year carbon saved and equivalent trees planted — making sustainability metrics tangible for end users.

Microsoft Elevate

Energy_Consumption_Optimizer.ipynb ☆ Changes will not be saved

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all Copy to Drive

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving household_power_consumption.csv to household_power_consumption.csv
```

LOAD & CLEAN DATA

```
import pandas as pd

df = pd.read_csv("household_power_consumption.csv", sep=',', low_memory=False)

df.replace('?', pd.NA, inplace=True)
df = df.dropna()

df['Global_active_power'] = df['Global_active_power'].astype(float)
df['Global_reactive_power'] = df['Global_reactive_power'].astype(float)
```

FEATURE ENGINEERING

Model 1: Linear Regression

[ ]

```
start = time.time()
lr = LinearRegression()
lr.fit(X_train, y_train)
lr_time = time.time() - start

lr_pred = lr.predict(X_test)
lr_mae = mean_absolute_error(y_test, lr_pred)
```

Model 2: Random Forest

[ ]

```
start = time.time()
rf = RandomForestRegressor(n_estimators=50)
rf.fit(X_train, y_train)
rf_time = time.time() - start

rf_pred = rf.predict(X_test)
rf_mae = mean_absolute_error(y_test, rf_pred)
```

[ ]

```
print("Linear Regression MAE:", lr_mae, "Training Time:", lr_time)
print("Random Forest MAE:", rf_mae, "Training Time:", rf_time)
```

CARBON FOOTPRINT CALCULATOR

```
[ ]
def calculate_carbon(units):
    emission_factor = 0.82  # kg CO2 per kWh
    return units * emission_factor
```

```
[ ]
sample_prediction = rf.predict([[10,15,6,0.3]])
carbon = calculate_carbon(sample_prediction[0])

print("Predicted Energy:", sample_prediction[0])
print("Estimated Carbon Emission:", carbon)
```

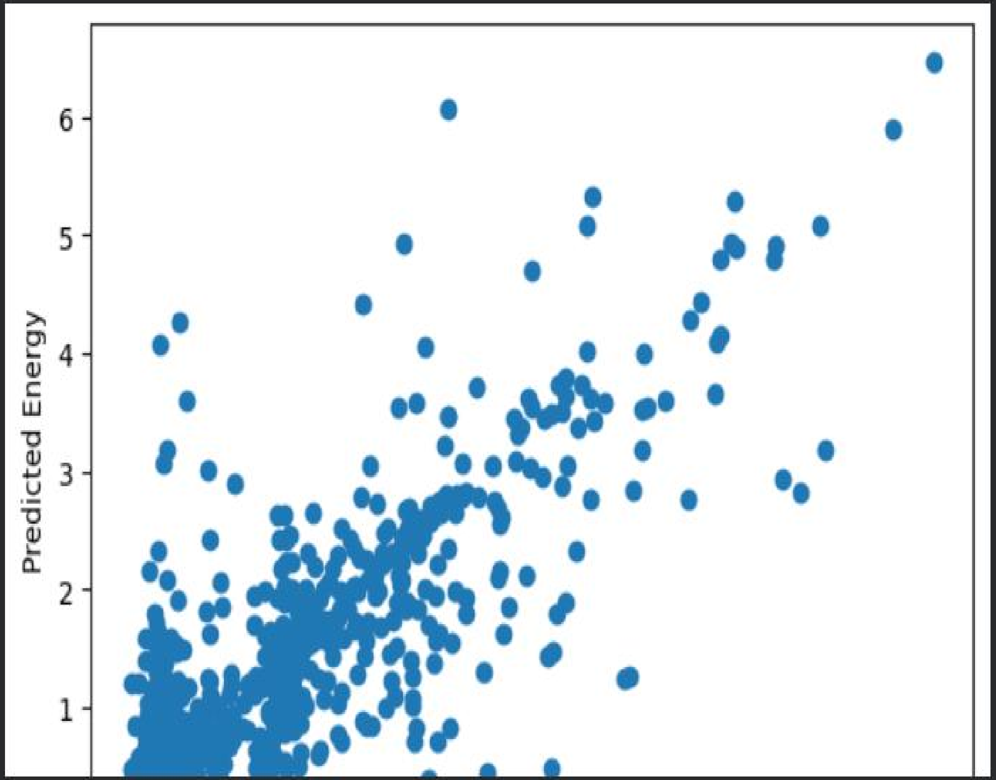
```
Predicted Energy: 0.9939933333333332
Estimated Carbon Emission: 0.8150745333333331
/usr/local/lib/python3.12/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature
  warnings.warn(
```

OPTIMIZATION ENGINE

```
[ ]
def optimize_energy(units):
    if units > 5:
        return "High usage detected. Reduce AC hours and heavy appliances."
    elif units > 3:
        return "Moderate usage. Switch to LED and avoid standby power."
```

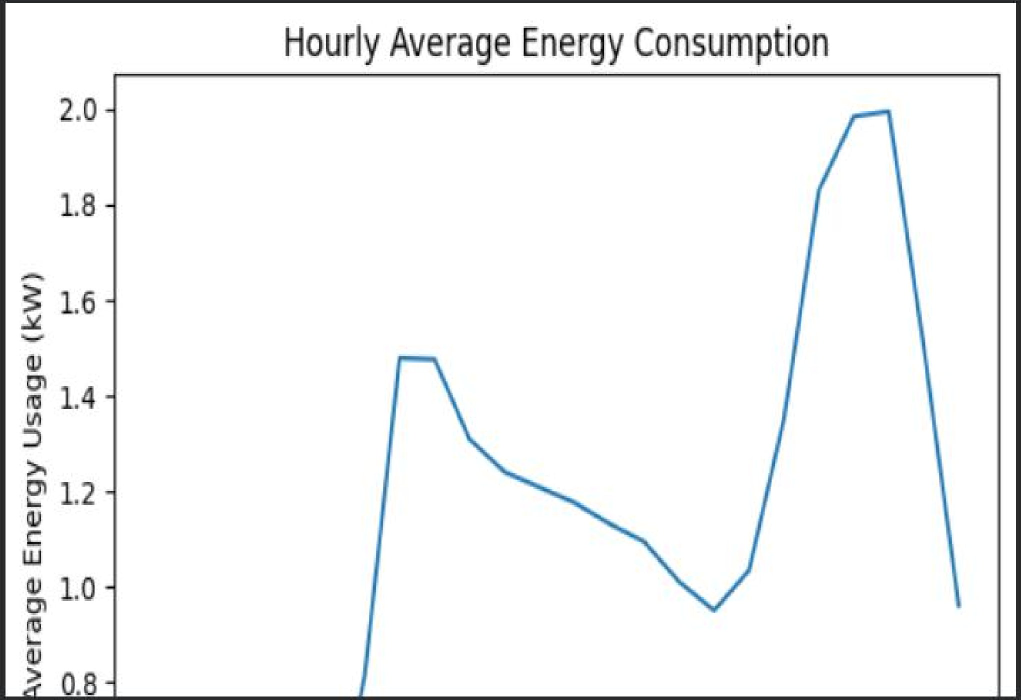
ADD VISUALIZATION

```
import matplotlib.pyplot as plt

plt.scatter(y_test[:1000], rf_pred[:1000])
plt.xlabel("Actual Energy")
plt.ylabel("Predicted Energy")
plt.show()
```

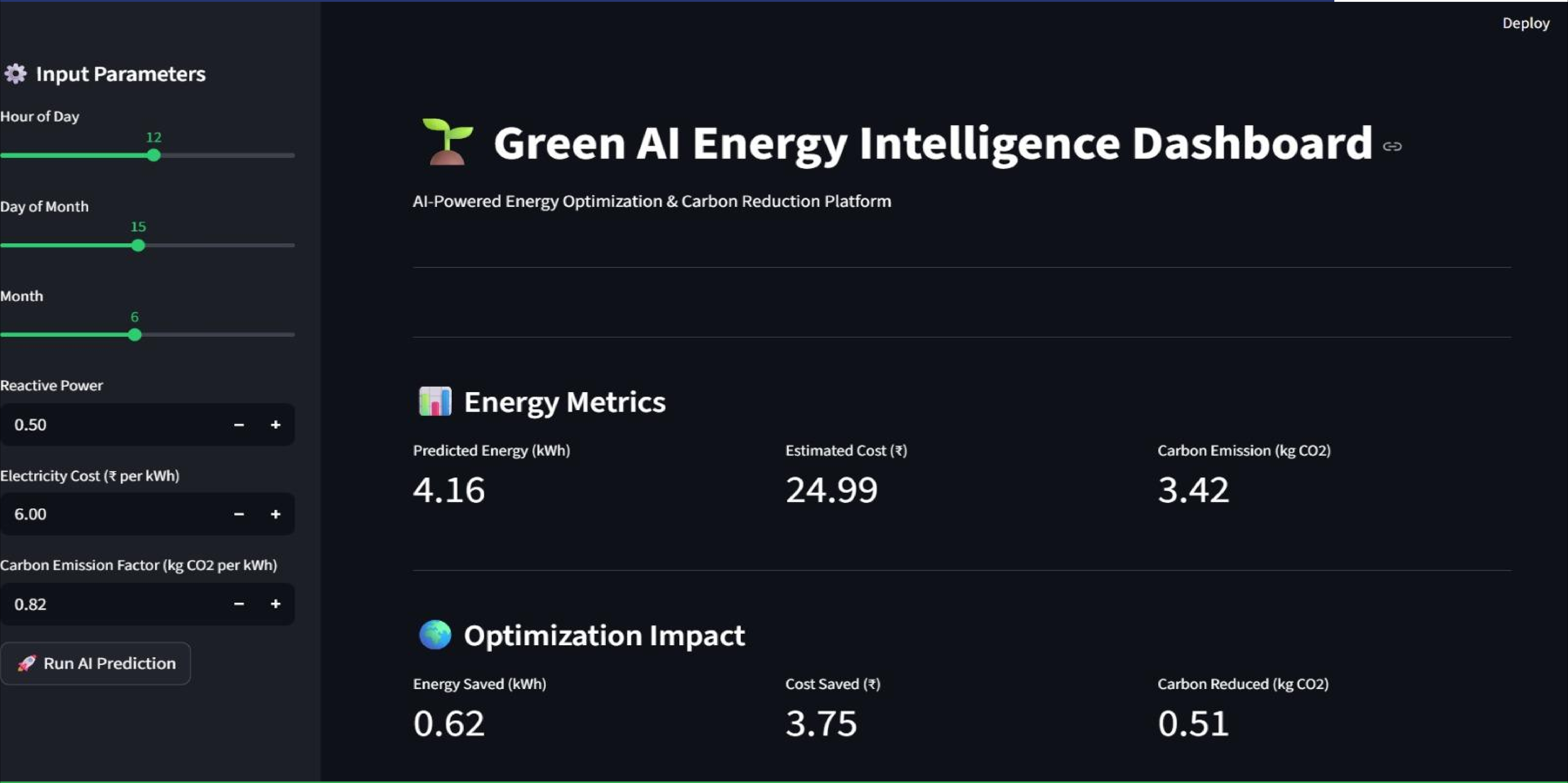


Hourly Average Energy Usage

```
hourly_avg = df.groupby('hour')['Global_active_power'].mean()

plt.figure()
plt.plot(hourly_avg.index, hourly_avg.values)
plt.xlabel("Hour of Day")
plt.ylabel("Average Energy Usage (kW)")
plt.title("Hourly Average Energy Consumption")
plt.show()
```



Microsoft Elevate



Microsoft Elevate

Deploy

Input Parameters

Hour of Day

12

Day of Month

15

Month

6

Reactive Power

0.50

Electricity Cost (₹ per kWh)

6.00

Carbon Emission Factor (kg CO2 per kWh)

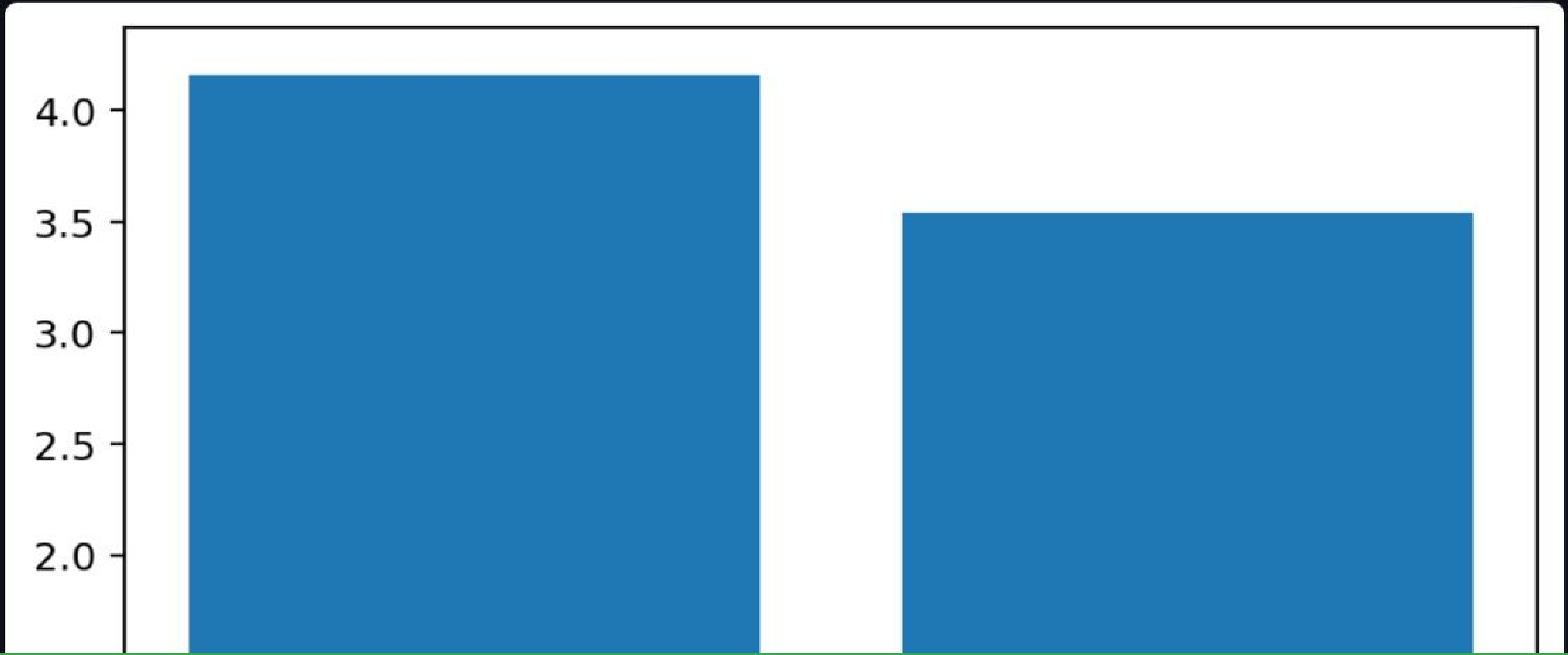
0.82

Run AI Prediction

Energy Category

Medium Energy Consumer

Energy Comparison



Microsoft Elevate

Deploy

Input Parameters

Hour of Day

12

Day of Month

15

Month

6

Reactive Power

0.50

Electricity Cost (₹ per kWh)

6.00

Carbon Emission Factor (kg CO2 per kWh)

0.82

Run AI Prediction

Smart 12-Month Projection

Monthly Efficiency Improvement (%)

5

Microsoft Elevate

Deploy

⚙️ Input Parameters

Hour of Day

12

Day of Month

15

Month

6

Reactive Power

0.50

Electricity Cost (₹ per kWh)

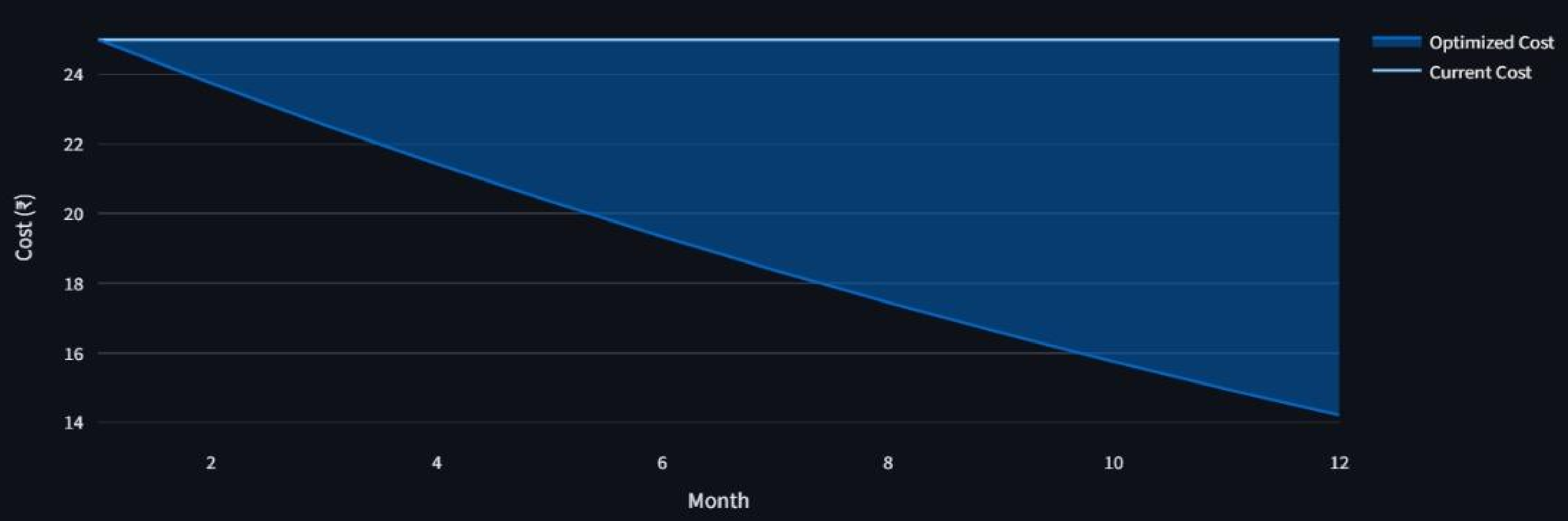
6.00

Carbon Emission Factor (kg CO2 per kWh)

0.82

🚀 Run AI Prediction

💰 Cost Projection



🌍 Carbon & Environmental Impact

🌿 1-Year Carbon Saved (kg CO2)

9.59

🌳 Equivalent Trees Planted

Microsoft Elevate

⚙️ Input Parameters

Hour of Day

12

Day of Month

15

Month

6

Reactive Power

0.50 − +

Electricity Cost (₹ per kWh)

6.00 − +

Carbon Emission Factor (kg CO2 per kWh)

0.82 − +

🚀 Run AI Prediction

🌍 Carbon & Environmental Impact

🌿 1-Year Carbon Saved (kg CO2)

9.59

🌳 Equivalent Trees Planted

0.5

📥 Download AI Report

📥 Download Report (CSV)

Green AI Internship Project | Energy Intelligence Platform | Built with Streamlit

CONCLUSION:

- The Green AI Energy Intelligence Platform successfully demonstrates how machine learning can be applied to real-world energy optimization — reducing consumption, cost, and carbon emissions simultaneously.

- The Random Forest model provides accurate energy predictions with a low MAE of 0.329, enabling reliable forecasting for household and industrial applications.

- The integrated carbon footprint calculator and optimization engine make sustainability actionable — translating raw data into clear environmental impact metrics.

- The Streamlit-based dashboard provides an intuitive, interactive interface for non-technical users to explore predictions, visualize trends, and download AI-generated reports.

- The project showcases end-to-end ML pipeline skills — from data cleaning and feature engineering to model training, evaluation, visualization, and deployment as a web application.

FUTURE SCOPE:

- **IoT Integration:** Connect with smart meters and IoT sensors to enable real-time energy monitoring and live predictions directly from household appliances.
- **Deep Learning Models:** Implement LSTM and Transformer-based time-series models for more accurate long-term energy consumption forecasting.
- **Appliance-Level Detection:** Use Non-Intrusive Load Monitoring (NILM) to disaggregate total energy usage into individual appliance consumption patterns.
- **Multi-User Dashboard:** Add user authentication and personalized dashboards to track individual household or building-level energy performance over time.
- **Renewable Energy Integration:** Incorporate solar panel output prediction and battery storage optimization to maximize green energy utilization.
- **Carbon Credit Marketplace:** Build a system to convert verified carbon savings into tradeable carbon credits, incentivizing energy-efficient behavior.

REFERENCES:

- GUCI Machine Learning Repository — Household Electric Power Consumption Dataset — https://archive.ics.uci.edu/dataset
- Scikit-learn Documentation — Random Forest Regressor — https://scikit-learn.org
- Streamlit Official Documentation — https://docs.streamlit.io
- SHAP (SHapley Additive exPlanations) — Lundberg & Lee (2017) — https://github.com/shap/shap
- Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5–32 — foundational paper on Random Forest algorithm.
- IEA (International Energy Agency). World Energy Outlook 2024 — global energy consumption and carbon emission benchmarks.
- GitHub Link: https://github.com/Suvasini911/StudyPal-AI

Thank You

For Your Valuable Time & Attention