Battery-Analysis-ML-Dashboard

A machine learning-based dashboard for predicting battery capacity using features like ambient temperature and resistance. It includes interactive visualizations, KNN and Random Forest models for predictions, and performance metrics such as MSE and R^2 to evaluate model accuracy.

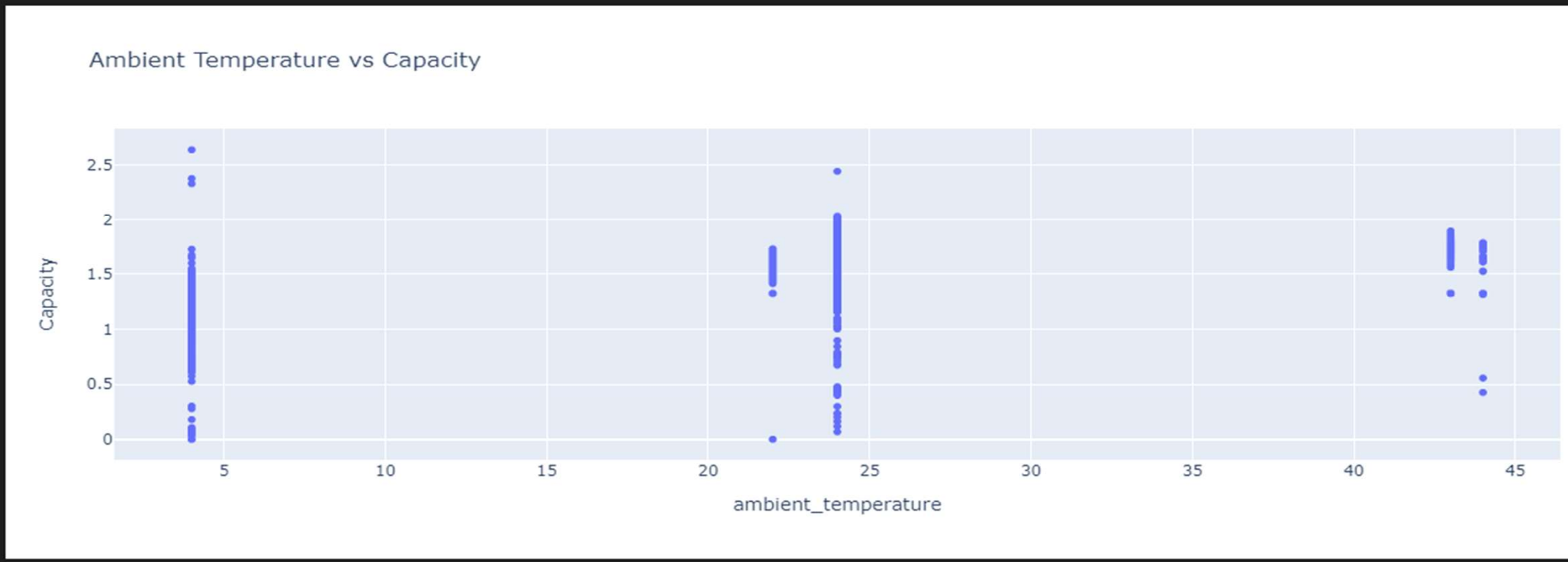
File Edit Selection View Go Run ··· Final_project

final_project_dash.ipynb

final_project_dash.ipynb > ...

+ Code + Markdown | Run All Restart Clear All Outputs | Jupyter Variables Outline ··· base (P

Battery Data Dashboard



⊗ 0 ⚠ 0 0 Reactivating terminals...

Cell 2 of 15 Go L

Type here to search

ENG 09 18-12

- The graph displayed is a **scatter plot** that visualizes the relationship between **Ambient Temperature** (x-axis) and **Capacity** (y-axis). Here's an analysis:
- **Before Data Cleaning:**
- **Observations:**
  - The data points are scattered but **clustered** in three distinct temperature ranges: **5°C, 25°C, and 45°C**.
  - Within each cluster, the **capacity values** (y-axis) are not well-aligned and appear noisy.
  - There are **outliers**, particularly at certain capacity values, which are inconsistent or extreme.
  - This suggests **missing values, duplicates, or erroneous entries** before cleaning.
- **After Data Cleaning :**
- **Key Improvements After Data Cleaning:**
  - **Outlier Removal**:
    - Extreme or unrealistic capacity values outside the expected range will be removed.
  - **Duplicate or Noisy Data Handling**:
    - Repeated values at the same temperature might be aggregated or filtered.
  - **Improved Structure**:
    - Data clusters would appear **smoother** with fewer irregular points.
    - Relationships between temperature and capacity would be clearer.
- **Conclusion:**
- The scatter plot **before data cleaning** shows data irregularities (noise and outliers).
- **After cleaning**, this graph will show better alignment of the data points, reduced outliers, and clearer trends, making it ready for model training.


File Edit Selection View Go Run
Final_project
final_project_dash.ipynb
final_project_dash.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Jupyter Variables Outline
base (Python 3.11.7)
Box Plot of Capacity
Capacity
2.5
2
1.5
1
0.5
0
Violin Plot of Capacity
Capacity
2.5
2
1.5
1
0.5
0
Callbacks
0 Errors
Server
Python
Reactivating terminals...
Cell 2 of 15
Go Live
Type here to search
ENG
09:28
18-12-2024

1. **Box Plot of Capacity** (on the left):
    1. The **box plot** summarizes the distribution of the "Capacity" variable.
    2. The **box** represents the **interquartile range (IQR)**, which contains the middle 50% of the data (Q1 to Q3).
    3. The horizontal line inside the box represents the **median**.
    4. The **whiskers** show the range of data (excluding outliers).
    5. Points **outside the whiskers** (dots) are **outliers**, indicating unusual battery capacity values.
    6. This graph highlights the presence of outliers above the upper range (beyond Q3).
2. **Violin Plot of Capacity** (on the right):
    1. The **violin plot** combines a box plot with a **kernel density estimate (KDE)**, showing the data distribution in a smooth, mirrored fashion.
    2. The **width** of the violin at different points represents the density of data.
    3. In this plot, you can see a concentration of battery capacity values around a specific range (close to 1.5).
    4. The tail (upper and lower) also indicates a few extreme values.

- **Observations**:
- The majority of battery capacities seem clustered near a central range.
- There are a few outliers with significantly larger capacity values.
- Both plots indicate a similar pattern, but the violin plot provides a better visualization of the density of values.
- These graphs help analyze the data distribution, spot outliers, and understand the variation in battery capacity.

File Edit Selection View Go Run ··· Final_project

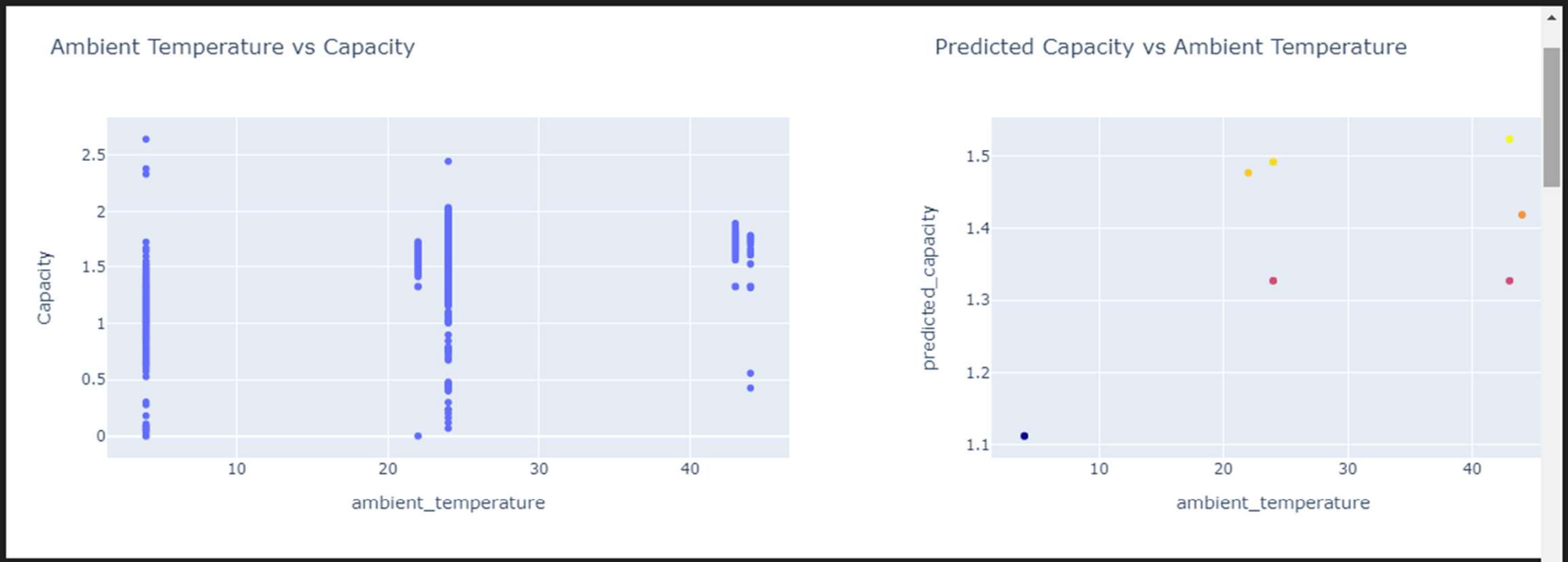
final_project_dash.ipynb

final_project_dash.ipynb > ...

+ Code + Markdown | Run All Restart Clear All Outputs | Jupyter Variables Outline ··· base (Python 3.11.7)

```
        print(f"Error running server: {e}")
```

[12] ✓ 4.6s Python



Actual vs Predicted Capacity

⊗ 0 ⚠ 0 0 Reactivating terminals... Cell 2 of 15 Go Live

Type here to search ENG 09:26 18-12-2024

- **Graph 1: Ambient Temperature vs Capacity (Left Plot)**This plot shows the relationship between **Ambient Temperature** and **Capacity** after **data cleaning**.

- **Observations**:
  - The points are better defined and cleaner, indicating **noise removal** or handling of missing/incorrect values.
  - There are still clusters at **specific temperature values** (e.g., 5°C, 25°C, 45°C), suggesting the data likely represents operational conditions at these temperatures.
  - Some variability in capacity remains, but the data appears **organized and cleaner** compared to raw data

- **Predicted Capacity vs Ambient Temperature**
- This graph shows the **predicted capacity** after applying a **machine learning or regression model**.

- **Observations**:
- The model has **smoothed out** the capacity values, showing a clearer trend or relationship with temperature.
- The **spread of predictions** is reduced, indicating that the model generalizes the patterns in the cleaned data.
- Different **colors** in the graph might represent different temperature ranges, predicted categories, or model outputs for specific conditions.

File Edit Selection View Go Run ···

Final_project

final_project_dash.ipynb

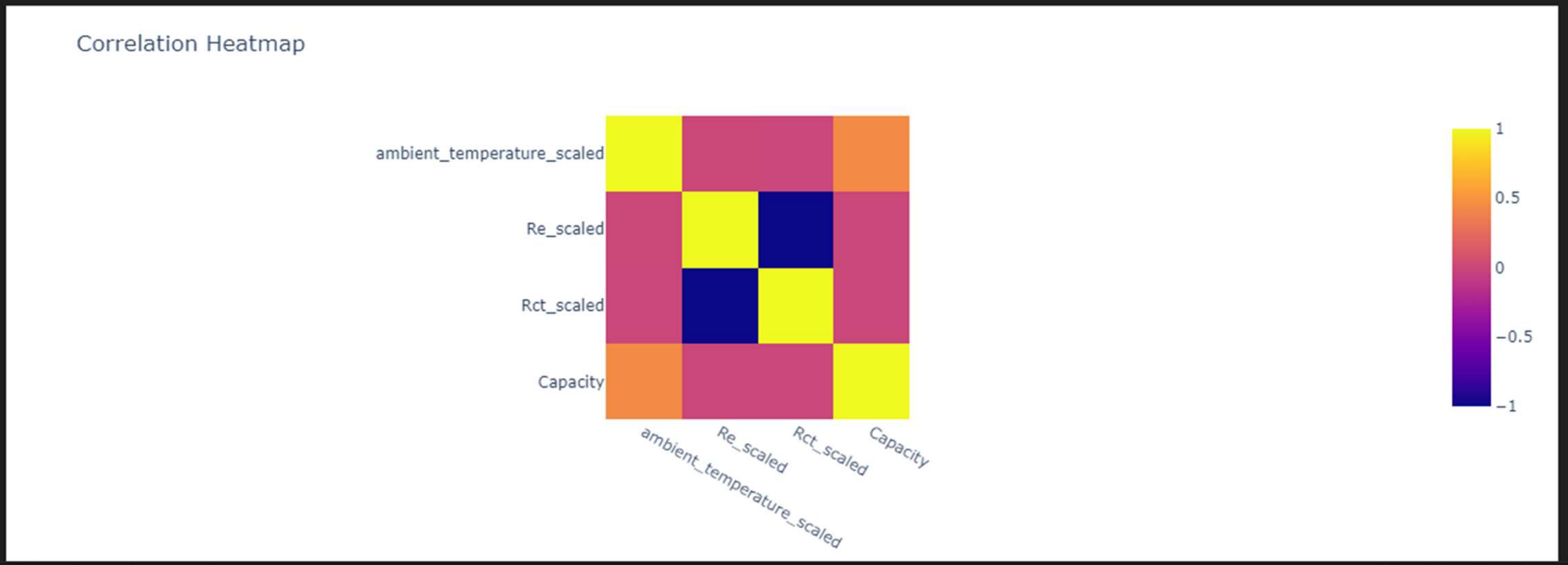
final_project_dash.ipynb > ...

+ Code + Markdown | Run All Restart Clear All Outputs | Jupyter Variables Outline ···

base (Python 3.11.7)

```python
# Run the app
if __name__ == '__main__':
    try:
        app.run_server(debug=True)
    except Exception as e:
        print(f"Error running server: {e}")
```

[12] ✓ 4.6s Python



⊗ 0 ⚠ 0 0 Reactivating terminals...

Cell 2 of 15 Go Live

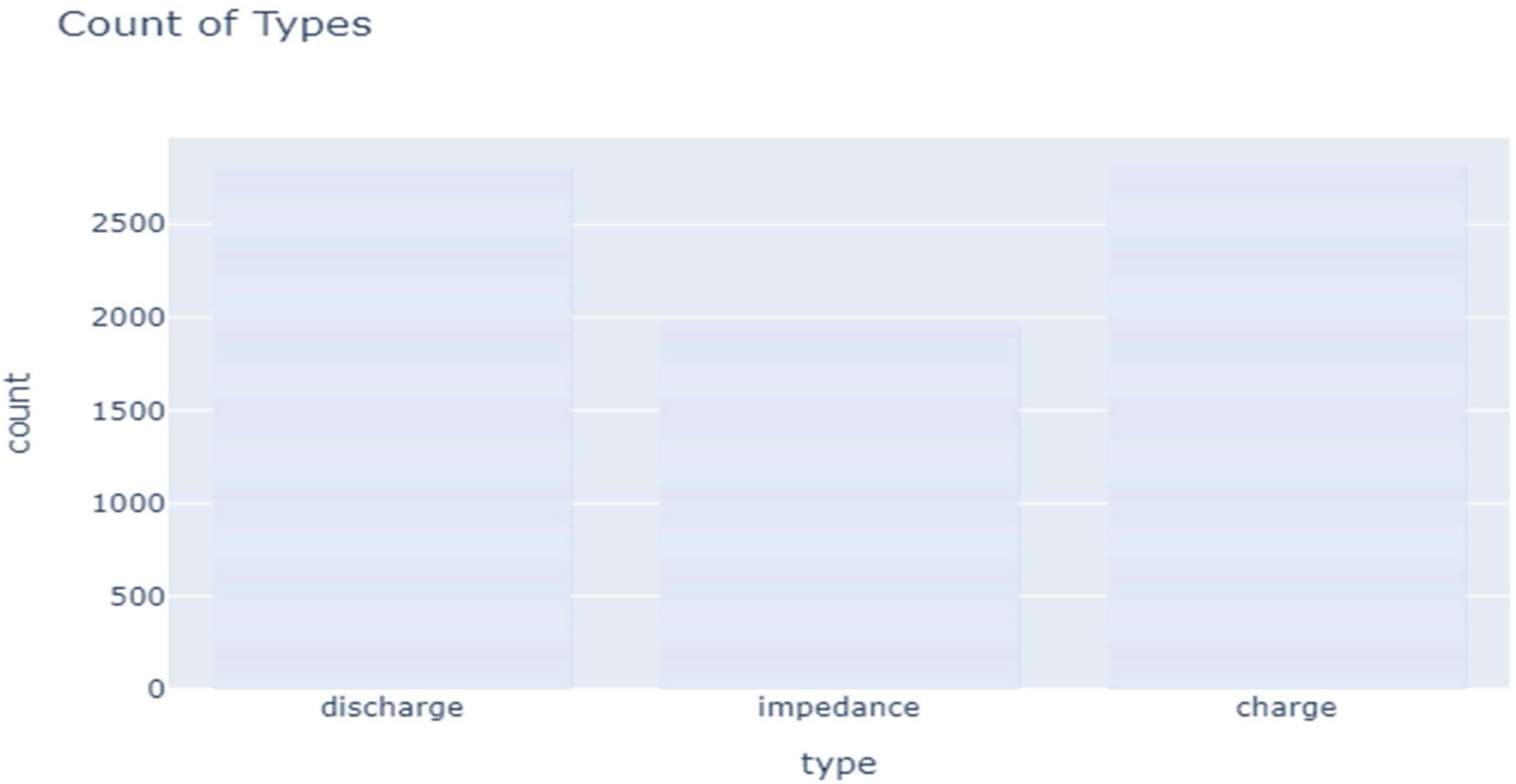
Type here to search

ENG 09:29 18-12-2024

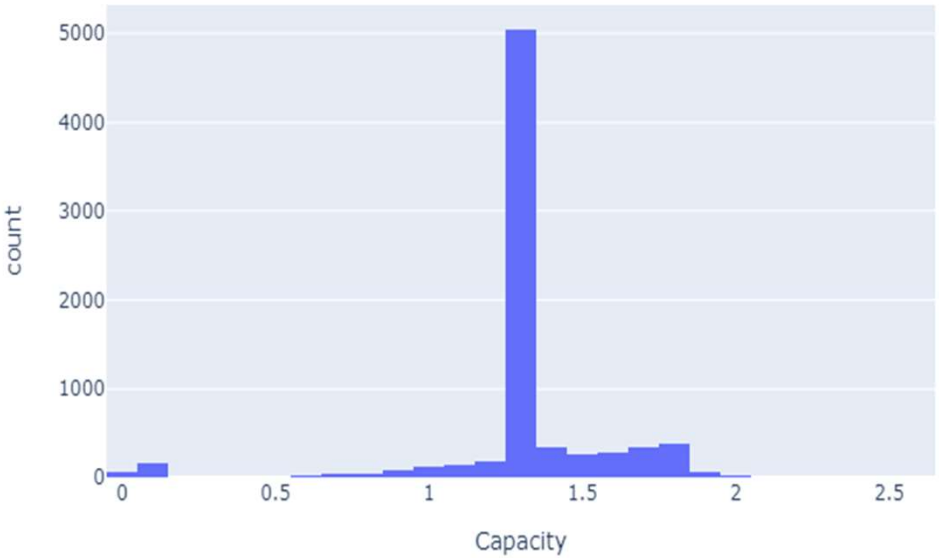
- # Heatmap Interpretation:
- The **Correlation Heatmap** shows the relationships between features:
- ambient_temperature_scaled
- Re_scaled (scaled resistance)
- Rct_scaled (another resistance parameter)
- Capacity (battery capacity)

- # Observations:
- **Color Representation:**
- Yellow: Strong positive correlation (~1).
- Dark Blue: Strong negative correlation (~-1).
- Pink/Purple: Neutral or weak correlation (~0)

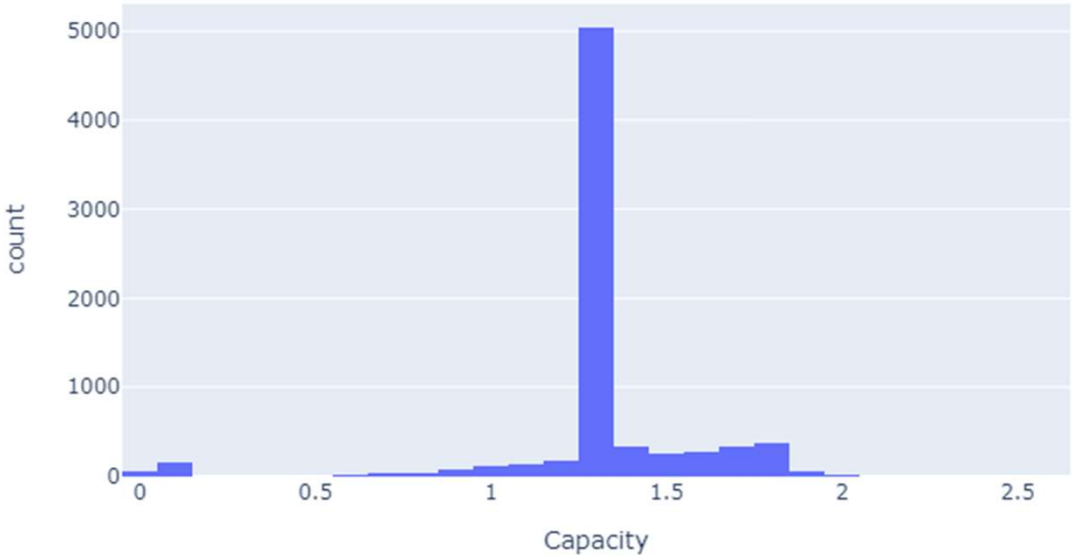
- **Key Insights:**
- Re_scaled and Rct_scaled have a strong negative correlation with each other (dark blue).
- Capacity shows a positive correlation with ambient_temperature_scaled (yellowish hue).
- Capacity has weaker correlations with other parameters (indicated by pink regions).
- This suggests that ambient_temperature_scaled plays a more significant role in predicting Capacity compared to other features

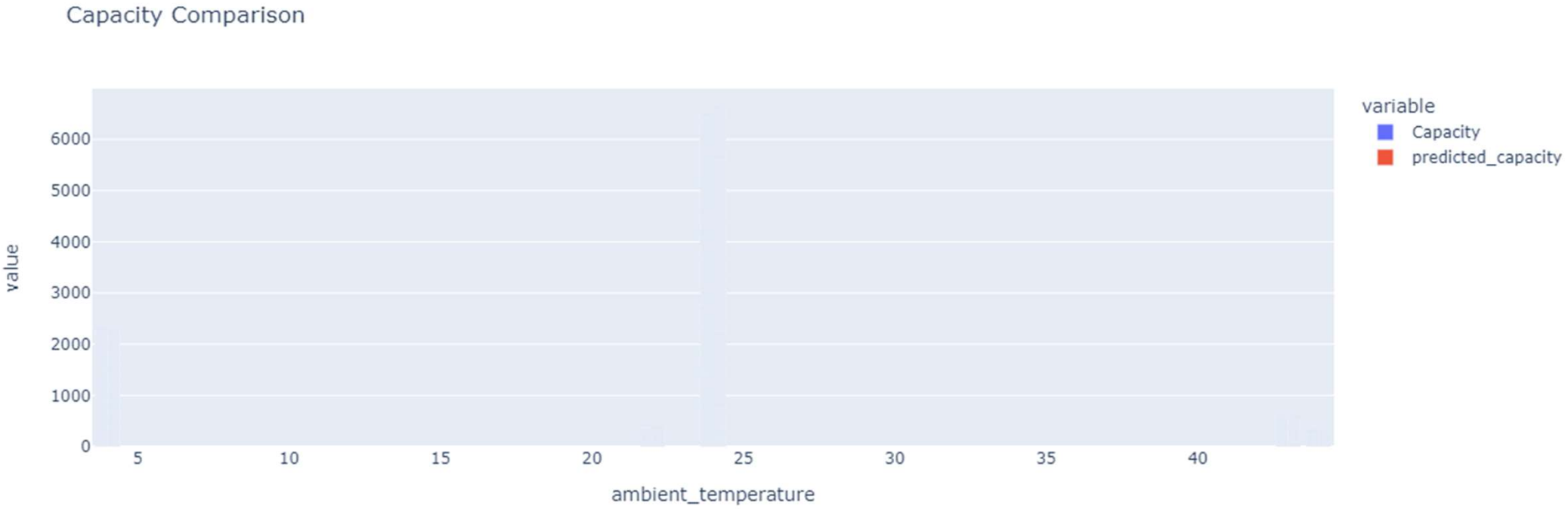

Count of Types
count
2500
2000
1500
1000
500
0
discharge
impedance
charge
type

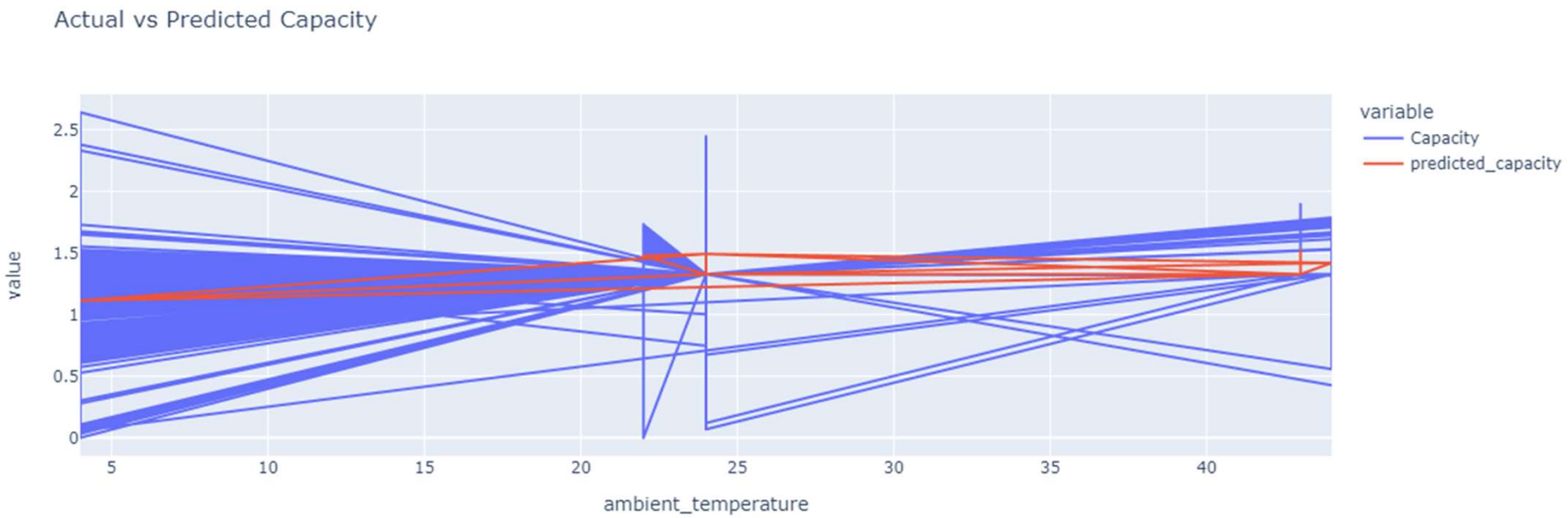
Distribution of Capacity

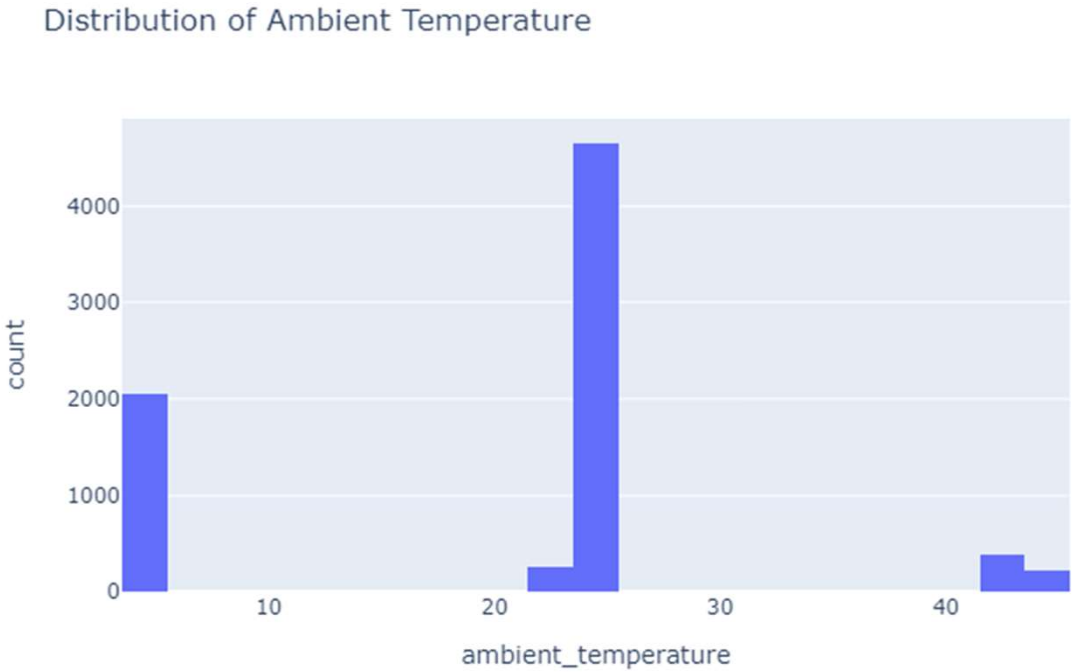


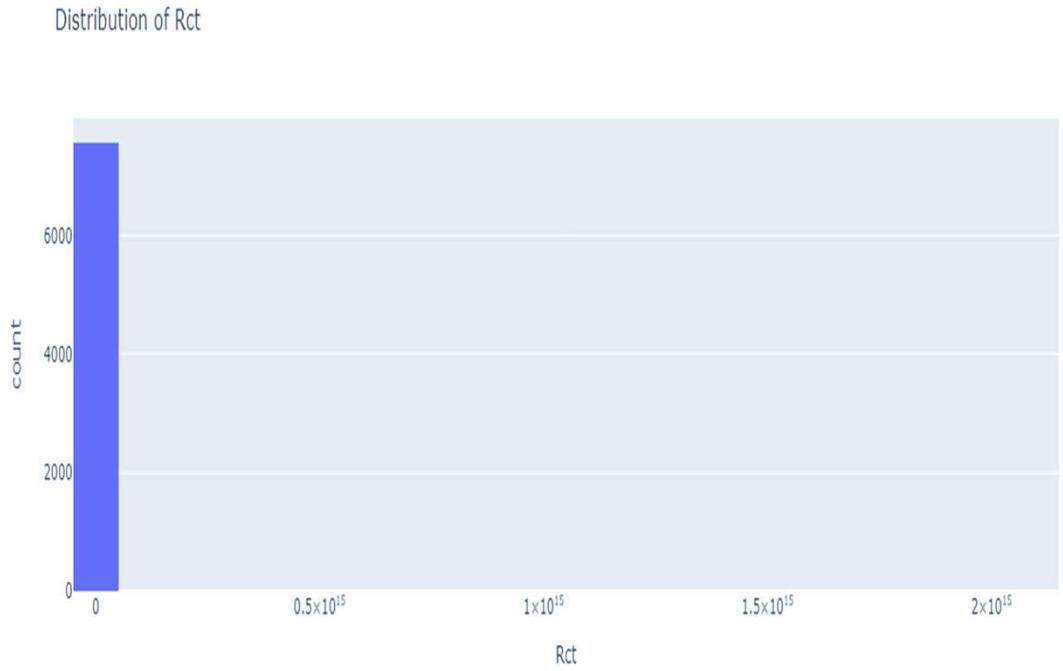
Distribution of Capacity

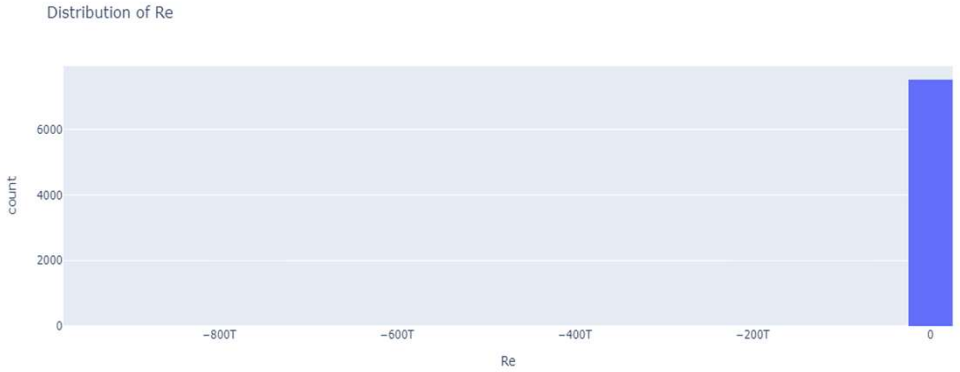


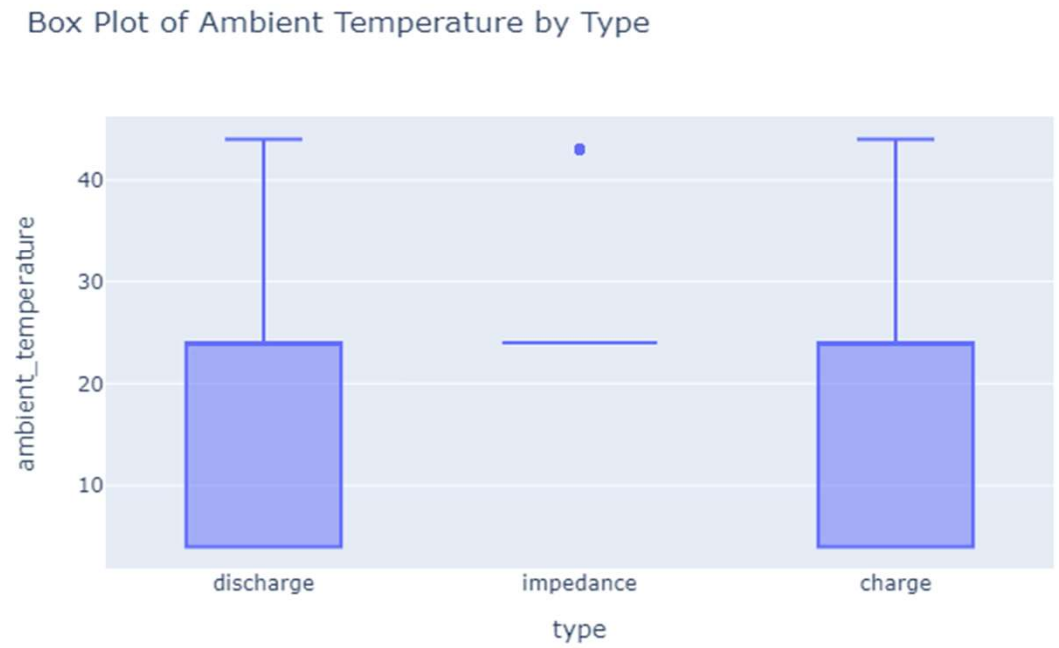

Capacity Comparison
value
6000
5000
4000
3000
2000
1000
0
5
10
15
20
25
30
35
40
ambient_temperature
variable
Capacity
predicted_capacity

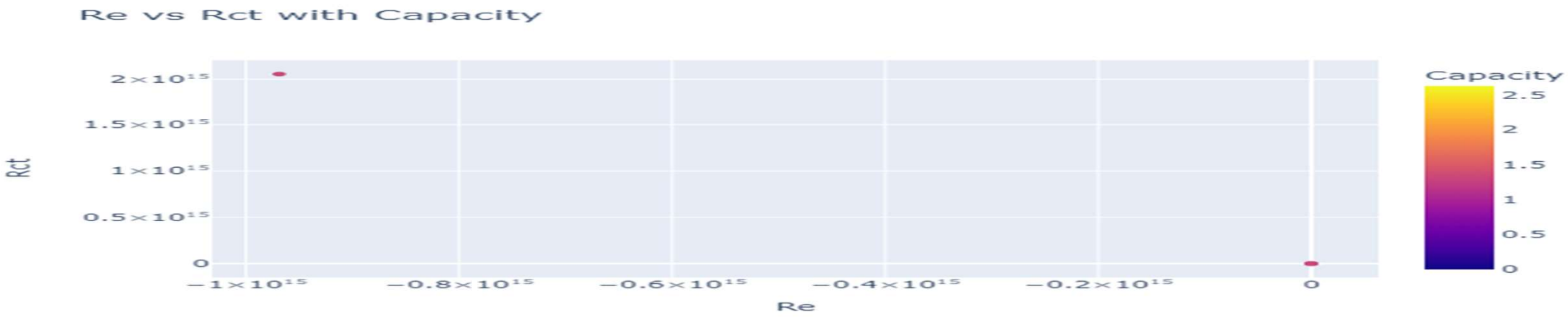

Actual vs Predicted Capacity
value
2.5
2
1.5
1
0.5
0
5
10
15
20
25
30
35
40
ambient_temperature
variable
Capacity
predicted_capacity


Distribution of Ambient Temperature
count
4000
3000
2000
1000
0
10
20
30
40
ambient_temperature


Distribution of Rct
count
6000
4000
2000
0
0
0.5×10^15
1×10^15
1.5×10^15
2×10^15
Rct


Distribution of Re
count
6000
4000
2000
0
−800T
−600T
−400T
−200T
0
Re


Box Plot of Ambient Temperature by Type
ambient_temperature
40
30
20
10
discharge
impedance
charge
type


Re vs Rct with Capacity
Rct
2×10¹⁵
1.5×10¹⁵
1×10¹⁵
0.5×10¹⁵
0
−1×10¹⁵
−0.8×10¹⁵
−0.6×10¹⁵
−0.4×10¹⁵
−0.2×10¹⁵
0
Re
Capacity
2.5
2
1.5
1
0.5
0

- **Final Remarks:**
- This dashboard effectively combines machine learning predictions with rich visualizations, providing actionable insights into battery capacity trends and model performance. While the model achieves an MSE of **0.0614**, further optimizations can enhance predictive accuracy for real-world applications.