Visualization Tool for Electric Vehicle Charge and Range Analysis

# 1. Project Overview

This project focuses on visualizing and analyzing electric vehicle (EV) data related to battery charge levels, driving range, and efficiency. The objective is to build an interactive tool for understanding EV performance under different conditions using data visualizations.

# 2. Project Goals

- Analyze EV charging patterns

- Visualize battery level, range, and efficiency trends

- Provide interactive tools for filtering based on model, trip duration, temperature, etc.

- Assist in decision-making for EV users and manufacturers

# 3. Tools and Technologies

| Tool | Purpose |

|--------------|----------------------------------|

| Python | Backend programming |

| Pandas | Data analysis |

| Matplotlib / Seaborn / Plotly | Data visualization |

| Streamlit / Flask | Frontend UI (interactive dashboards) |

| CSV / JSON | Data source format |

# 4. Dataset Description

| Column Name | Description |

|---------------------|------------------------------------------|

| timestamp | Time of the log |

| battery\_level | Battery percentage (0-100%) |

| range\_km | Estimated range in kilometers |

| temperature | External temperature during log |

| charging\_status | Charging (Yes/No) |

| vehicle\_model | EV model (e.g., Tesla Model 3) |

# 5. Step-by-Step Code Logic

Step 1: Import Libraries

```

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import plotly.express as px

```

Step 2: Load the Dataset

```

df = pd.read\_csv('ev\_data.csv')

df['timestamp'] = pd.to\_datetime(df['timestamp'])

```

Step 3: Preprocess the Data

```

df.dropna(inplace=True)

df.sort\_values(by='timestamp', inplace=True)

df['hour'] = df['timestamp'].dt.hour

df['date'] = df['timestamp'].dt.date

```

Step 4: Visualizations

➤ Battery Level Over Time:

```

plt.figure(figsize=(12, 6))

sns.lineplot(data=df, x='timestamp', y='battery\_level')

plt.title('Battery Level Over Time')

plt.show()

```

➤ Range vs. Temperature:

```

sns.scatterplot(x='temperature', y='range\_km', hue='vehicle\_model', data=df)

plt.title('Range vs Temperature')

plt.show()

```

➤ Interactive Plot (Plotly):

```

fig = px.line(df, x='timestamp', y='range\_km', color='vehicle\_model', title='Range Over Time')

fig.show()

```

Step 5: Build Interactive Dashboard (Streamlit Example)

```

import streamlit as st

st.title("EV Charge & Range Analysis")

selected\_model = st.selectbox("Choose Vehicle Model", df['vehicle\_model'].unique())

filtered\_df = df[df['vehicle\_model'] == selected\_model]

st.line\_chart(filtered\_df.set\_index('timestamp')['battery\_level'])

st.line\_chart(filtered\_df.set\_index('timestamp')['range\_km'])

```

# 6. Result Screenshots (optional)

- Line chart of battery level and range

- Dropdown to filter by model

- Range variation over external temperature

# 7. Challenges Faced

- Handling missing sensor values

- Combining multiple logs from different EV models

- Syncing timestamps during visualization

# 8. Final Outcome

- A working interactive tool to visualize and explore electric vehicle data

- Ability to filter data by time, model, or condition

- Generated insights into charging patterns and range drop with temperature

# 9. Future Enhancements

- Integrate real-time EV telemetry data

- Add map visualization using GPS coordinates

- Predict future range using ML models

- Enable export of visualizations

# 10. Folder Structure

```

ev\_visualization\_tool/

├── ev\_data.csv

├── app.py / dashboard.py

├── templates/

│ ├── index.html

│ └── result.html

├── static/

│ └── plots/

└── README.md

```