EV BATTERY - PYTHON PROJECTS - AI & CYBERSECURITY

SUDARSHANA KARKALA | EV.ENGINEER | +91 9845561518 | carsoftwaresystems@gmail.com | CAR SOFTWARE SYSTEMS (.com)

BATTERY TEMPERATURE MONITORING SYSTEM

Goal : Read temperature data, analyse trends, and detect overheating.
Concepts: File handling, NumPy, Pandas, Matplotlib

Tasks:

- Read a CSV file containing battery temperature data
- Calculate average, max, and min temperatures
- Plot a temperature trend graph using Matplotlib
- Detect overheating conditions (e.g., alert if temp > 60°C)

Outcome: Basic battery monitoring using Python

BATTERY VOLTAGE & CURRENT ANALYSIS

Goal: Analyse voltage & current data to detect anomalies.
Concepts: Pandas, Data Visualisation, Time-Series Analysis

Tasks:

- Load battery voltage & current datasets
- Identify voltage drops and current spikes
- Plot Voltage vs. Time & Current vs. Time
- Set a rule: Alert if voltage drops below a threshold

Outcome: Detect battery performance issues

STATE OF CHARGE (SOC) ESTIMATION

Goal: Estimate battery SOC using voltage and current data.
Concepts: Numerical computing, Basic Machine Learning

Tasks:

- Load historical battery data (Voltage, Current, SOC)
- Train a simple regression model to predict SOC
- Validate results using test data
- Display real-time SOC values for a given input

Outcome: SOC estimation using Python

EV BATTERY HEALTH PREDICTION (AI MINI PROJECT)

Goal: Use AI to predict battery degradation over time.
Concepts: Machine Learning, Data Science

Tasks:

- Load battery charge-discharge cycle data
- Identify patterns in battery degradation
- Train an ML model (Scikit-learn) to predict Remaining Useful Life (RUL)
- Visualise predictions with graphs

Outcome: AI-based battery health prediction

REAL-TIME BATTERY MONITORING WITH IOT (ADVANCED)

Goal: Collect real-time battery data using IoT (optional).
Concepts: Python, MQTT, IoT Sensor Integration

Tasks:

- Connect a temperature sensor (DHT11) or voltage sensor
- Use Raspberry Pi / ESP8266 to collect data
- Send data via MQTT or HTTP to Python
- Analyse and store data in a database

Outcome: Real-time battery health monitoring

INTRUSION DETECTION IN BATTERY MANAGEMENT SYSTEM (BMS)

Goal: Detect anomalous activities (hacking attempts, data tampering, or unauthorised access) in an EV Battery Management System (BMS) using Python.

Concepts Used:

- Log Analysis & Data Forensics
- Anomaly Detection (Machine Learning)
- Cybersecurity Threat Detection

Project Overview: The Battery Management System (BMS) logs critical parameters:

- Voltage, Current, Temperature
- State of Charge (SOC), State of Health (SOH)
- Communication logs (CAN messages)

Potential Cyber Threats:

- **Spoofing Attack**: Fake voltage readings injected
- **Man-in-the-Middle Attack:** SOC data modified
- **Malware in BMS**: Unauthorised data manipulation

Expected Outcomes

- Build a Battery Intrusion Detection System (IDS)
- Detect cyber attacks on BMS data
- Train an ML model to differentiate between normal and attack conditions
- (Advanced) Secure BMS communication with encryption

IMPLEMENTATION STEPS

STEP 1: Collect Battery Data Logs (or Use Sample Data)

- Use a CSV file containing battery logs with timestamps
- Add a column for intrusion detection labels (Normal / Attack)

STEP 2: Analyse Normal vs. Anomalous Data

- Load the dataset using Pandas
- Visualise voltage/current variations using Matplotlib
- Identify unexpected spikes, drops, or inconsistent SOC values

STEP 3: Implement an Anomaly Detection Model

- Use Scikit-Learn to train an ML model for intrusion detection
- Algorithms: Isolation Forest, Random Forest, or Logistic Regression
- Train model on normal vs. attack data samples
- Detect real-time anomalies from live battery logs

STEP 4: Real-Time Intrusion Detection Simulation

- Simulate incoming battery data (live stream using Python)
- Detect unauthorised activities and trigger alerts
- Implement logging system to save security breach attempts

STEP 5: (Advanced) Secure Battery Data with Encryption

- Use AES Encryption (Python pycryptodome module)
- Encrypt critical BMS data before transmission
- Ensure only authorised systems can decrypt it