Enhancing Battery State of Health Estimation using Machine Learning Techniques

An undergraduate thesis submitted by

Md. Sadman Sakib (180021126)

Maryam Jamila Busra (180021301)

Redwan Ahmed (180021329)

in consideration of the partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE

in

ELECTRICAL AND ELECTRONIC ENGINEERING

Department of Electrical and Electronic Engineering

Islamic University of Technology (IUT)

June 2023

DECLARATION

I do hereby solemnly declare that the research work presented in this thesis has been carried out by me and has not been previously submitted to any other University / Institute / Organization for an academic qualification / certificate / diploma or degree.

Md. Sadman Sakib
Student ID: 180021126
Date: 05 June, 2023

Maryam Jamila Busra
Student ID: 180021301
Date: 05 June, 2023

Redwan Ahmed
Student ID: 180021329
Date: 05 June, 2023

APPROVAL

The thesis titled ”Enhancing Battery SoH Estimation using Advanced Machine Learning Techniques” submitted by Md. Sadman Sakib (180021126), Maryam Jamila Busra (180021301), and Redwan Ahmed (180021329) has been found as satisfactory and accepted as partial fulfillment of the requirement for the degree of Bachelor of Science in Electrical and Electronic Engineering on 5th June, 2023.

Approved by:

(Signature of the Supervisor)
Dr. Md. Ashraful Hoque
Professor
Department of Electrical and Electronic Engineering
Islamic University of Technology

ACKNOWLEDGEMENTS

First and foremost, we extend our deepest gratitude to Almighty Allah for bestowing upon us the perseverance and resilience to successfully accomplish this thesis work.

We are tremendously appreciative of the chance to collaborate with Prof. Dr. Md. Ashraful Hoque, Professor in the Department of Electrical and Electronic Engineering at IUT, as our excellent supervisor. As we started on this research adventure, his excellent guidance and insights pushed us in the correct direction, motivating us to dig deeper and generate groundbreaking findings. His unflinching support has been a guiding light for us throughout this journey, keeping us on course.

Additionally, we would like to convey our sincere gratitude to Mr. Mirza Muntasir Nishat, Assistant Professor in the Department of Electrical and Electronic Engineering at IUT, who served as our co-supervisor and generously gave of his time and effort to mentor us. He introduced us to the subject of the study when we were just beginning, and his ongoing guidance and criticism have been of great use. He has been a source of inspiration and assurance because of his capacity to elevate our spirits and provide answers to problems, regardless of how difficult they may be. We were able to carry out this research successfully thanks to his clear instructions.

The Authors
June 2023
Gazipur, Bangladesh

ABSTRACT

Electric vehicles (EVs) have gained significant popularity due to their improved performance and contribution to addressing climate change. However, the cost of EVs remains a challenge, with lithium-ion batteries being one of the most expensive components. A correct SoH estimation allows the quantification of the residual market value of the battery pack, enabling customers to sell the battery before it exceeds its end-of-life in the target application, while still ensuring safety and reliability for reuse in other domains. This paper proposes the use of four ensemble learning algorithms to estimate the SoH of Li-ion batteries. Our approach also incorporates measures to ensure the applicability of the methods in real-world applications. This study leverages two datasets comprising charge profiles of lithium-ion batteries to develop and evaluate the proposed models. By employing popular ensemble algorithms such as Random Forest, XGBoost, LightGBM, and CatBoost, alongside univariate and multivariate feature selection techniques, and an unsupervised anomaly detection technique, Isolation Forest, we address the challenge of precise battery SoH estimation. The results highlight the superior performance of LightGBM among the tested models using appropriate features selected by feature selection techniques. By enhancing the accuracy of SoH estimation for lithium-ion batteries, this work promotes consumer confidence in the adoption of electric vehicles and renewable energy sources.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

SoH State of Health

EV Electric Vehicle

LCO Lithium Cobalt Oxide

ML Machine Learning

ANN Artificial Neural Network

CNN Convolutional Neural Network

RNN Recurrent Neural Network

LSTM Long Short-Term Memory

SVM Support Vector Machine

GPR Gaussian Process Regressor

RF Random Forest

XGB eXtreme Gradient Boosting

LGBM Light Gradient Boosting Machine

RMSE Root Mean Squared Error

MAE Mean Absolute Error

R2 Coefficient of Determination

MAPE Mean Absolute Percentage Error

CVCT Constant Voltage Charge Time

CCCT Constant Current Charge Time

CHAPTER 1

INTRODUCTION

Lithium-ion batteries are the primary energy storage for electric vehicles, solar-based power systems and several other renewable energy sources. They play a key role in transitioning from a fossil fuel based economy to a green energy based economy with a view to reducing carbon emission. Lithium ion batteries are characterized by stable electrochemical properties, high energy density and superior lifetime in comparison with conventional nickel-cadmium, nickel-metal-hydride cells and lead-acid batteries [1]. However, these batteries are prone to capacity degradation over their lifetime, which poses significant challenges for battery management systems. Accurately estimating battery health, specifically the State of Health (SoH), is of paramount importance to mitigate battery failure risks [2], enable fast charging schemes tailored to different aging conditions, facilitate battery replacement decisions, and promote battery recycling and second-life usage.

State of Health of a Li-ion battery is defined as the current capacity compared to its nominal capacity. It quantifies the degradation or loss of capacity in a battery over time. SoH is typically expressed as a percentage, where 100% indicates a battery's full capacity and 0% represents complete failure or loss of capacity.

The State of Health (SoH) of a battery can be represented by the equation:

$$\text{SoH} = \left(\frac{C_i}{C_0}\right) \times 100\% \tag{1.1}$$

where, C_i represents the current capacity or remaining capacity of the battery and C_0 represents the nominal or rated capacity the battery.

SoH estimation is conducted following two main approaches- physical model based approach and data-driven method. The physics based models predict the state of health by calculating complex mathematical parameters of the electrochemical of the battery and analyzing the failure progressively. On the other hand, the data-driven models predict the state of health by feeding pre-acquired data to a machine learning model.

Various model-based methods including Coulumb Counting, Open circuit voltage, Particle Filter, Kalman Filter, Impedance Spectroscopy have been used for SoH estimation

[3, 4, 5, 6, 7]. Open circuit voltage, coulomb counting and impedance spectroscopy are used to directly measure the state of health in common scenarios. The OCV-based method for estimating the SoH of the battery is oriented on defining the SoH as a function of the OCV of the monitored battery. Electrochemical impedance spectroscopy (EIS) is a method which permits obtaining highly accurate impedance measurements of a battery cell over a large spectrum of frequencies at low currents [8]. Under the operating limit of voltage current and temperature, the voltage profile with the same charging current can reflect the aging of the battery [9]. OCV model can be used to perform battery SOH monitoring as it effectively captures aging information based on incremental capacity analysis (ICA) [10].

Data-driven methods, on the other hand, are dependent only on the necessary data, not as resource hungry as physics based methods and require less amount of time to process. Machine learning is a widely used data-driven method which can be applied in various domains of analysis including battery health prediction. It potently produces great results using practically acquired data and fewer computational resources.

In estimating the SoH of batteries using Machine Learning techniques, some challenges arise, including data availability, noise in captured data, and selecting the appropriate model. However, despite these challenges, Machine Learning remains a more practical approach for real-world implementation than physics-based or equivalent circuit-based methods. More insights on the literature on machine learning for SoH estimation is provided in Chapter 2.

Our research aims to improve the accuracy and reliability of battery SoH estimation through the evaluation of various ensemble algorithms and feature selection techniques. Doing so, we can optimize charging and discharging protocols to minimize battery stress and extend its lifespan. Our goal is to contribute to the development of more reliable battery technologies.

To guide the investigation, the following research questions will be addressed:

1. What are the difficulties with using ML techniques for online SoH estimation and how to address them in the most economic way?

2. What features should be used to train the model for real-life application?

By answering these research questions, the project aims to contribute to the body of knowledge surrounding SoH estimation.

This thesis follows a well-structured approach to achieve its goals. The project begins with a comprehensive introduction in Chapter 1, setting the context and providing an

overview of the research using both model-based and data driven techniques. Chapter 2 focuses on conducting a thorough literature review specially on data driven techniques to gain insights from existing studies and research findings. Chapter 3 gives a quick overview of the algorithms used and the reason for their usage. Chapter 4 presents the methodology employed, outlining the data collection, data preprocessing techniques, and hyperparameter optimization. In Chapter 5, the results obtained from the research are presented and discussed, shedding light on the effectiveness of the applied methods. Finally, Chapter 6 offers conclusive remarks, summarizing the findings, and highlighting the significance of the research outcomes.

CHAPTER 2

LITERATURE REVIEW

The challenge of accurately estimating the state of health of batteries is of paramount importance for the adoption of electric vehicles and renewable energy sources. In recent years, various data-driven approaches have been explored as potent solutions to address this issue. A wide array of machine learning algorithms, such as support vector machines, decision trees, random forest, neural networks, and other feature extraction methods, have been extensively applied in this domain.

In [11], to estimate battery SoH and SoC, significant features were extracted from discharge curves. They found out that polynomial regression models outperform other ML techniques. However, their obatined error was significant. Support vector regression (SVR) was used with optimized hyperparameters to estimate SoH, employing a differential evolution algorithm [12]. A fusion-type SoH estimation method was proposed, utilizing a novel model-based voltage construction method to reshape disturbance-free incremental capacity curves [13]. A uniform estimation framework was introduced to estimate SoH and optimize healthy features using charging voltage curves within a fixed range [14]. The approach involved a fixed-size least squares-support vector machine and genetic algorithm for parameter tuning. This allowed less computation intensity, high robustness and universality. An adaptive fusion algorithm was developed to estimate the state of charge and state of health of lithium-ion batteries. The method utilized an improved recursive least squares algorithm with a forgetting factor and least squares support vector machine for parameter identification and estimation [15] Also, an adaptive H-infinity filter algorithm was applied to predict the battery state of charge and to cope with uncertainty of model errors and prior noise evaluation.

Health indicators (HIs) were classified into measured and calculated variables, and different feature selection methods were compared [16]. Four noise reduction methods for HIs extraction and applied different feature selection methods, including filter-based, wrapper-based, and fusion-based approaches, to select subsets of HIs. The performance of four machine learning algorithms, namely artificial neural network, support vector machine, relevance vector machine, and Gaussian process regression, was compared. The results indicated

that the combination of the fusion-based selection method and Gaussian process regression yielded superior estimation performance in terms of both accuracy and computational efficiency.

A feature selection method for SoH estimation and a new training set design for the least squares support vector machine algorithm were proposed [17]. They selected the features theoretically, proved and then re-screened mathematically. A feature extraction method using generic discharging conditions was developed, and health indicators associated with battery capacity were identified [18]. A filtering technique is utilized to create smooth voltage curves under dynamic discharging settings, and a voltage partition method is used to acquire the discharge capacity differences of two cycles from non-monotonic or pulse discharge voltage curves. Gaussian process regression achieved the best performance among the compared algorithms, which included linear regression, support vector machine, and relevance vector machine. Two novel methods for estimating the health status of lithium-ion batteries were proposed using decision tree and random forest algorithms [19]. Four features were extracted related to actual capacity degradation from the data. An intelligent SoH estimation framework based on real-world data collected from electric vehicles was developed [20]. A feedforward neural network driven by the degradation index achieved effective SoH estimation and aging trend description. The estimation method is validated by the oneyear monitoring dataset from 700 vehicles with different driving mode.

Ensemble learning and gray relational analysis were utilized for feature correlation analysis, resulting in an accurate and stable SoH prediction model [21]. The XGBoost algorithm with accuracy correction by Markov chain was employed to improve the accuracy of SoH estimation [22]. Several features such as average voltage, voltage difference, current difference, and temperature difference were extracted, leading to improved accuracy compared to other algorithms such as RF, SVM, LR and KNN. Voltage, current, and temperature profiles were leveraged for SoH estimation using feedforward neural network, convolutional neural network, and long short-term memory models [23]. The multi-channel technique based on these profiles outperformed using only voltage profiles. Random forest was used for battery capacity estimation based on charging voltage and capacity measurements [24]. For feature selection, incremental capacity analysis was used. The model was solely based on signals, such as the measured current, voltage and time, that are available onboard during typical battery operation. The collected raw data can be directly fed into the trained model without any pre-processing, leading to a low computational cost.

Health indicators were extracted from battery current, voltage, and temperature data, and the correlation between health indicators and battery capacity degradation was quantified using grey relation analysis [25]. An online diagnosis method was proposed using a fixed-size least squares support vector machine with optimized parameters [26]. Short-term charging data was used to predict future voltage profiles and determine SoH. For parameter optimization, simulated annealing method was used. In [27], a novel weighted Gaussian Process Regression (GPR) approach was proposed for state of health (SoH) estimation. The method aimed to reduce the model's dependence on data through knowledge transfer. A squared exponential covariance function with a penalty mechanism was introduced to control the cross-battery knowledge transfer process. Experiments were conducted using battery cyclic aging data under different working conditions.

In [28], the authors suggested that the chi-square of battery voltages and the mean temperature could effectively reflect battery capacity loss. They utilized an ensemble algorithm composed of Extreme Learning Machine (ELM) and Long Short-Term Memory neural network to capture the underlying correspondence between SoH, mean temperature, and chi-square of battery voltages.

A Support Vector Regression approach was proposed for establishing a battery degradation model [29]. Battery capacity was estimated using partial incremental capacity curves, and advanced filter algorithms were applied to smooth the curves. Additionally, a peak fitting technique was used to decompose the smooth curves, and battery health features were extracted from the decomposed incremental capacity curves as training datasets. In [30], a least square support vector regression model with a polynomial kernel function was utilized for battery capacity estimation. Feature samples of battery health state were extracted from the battery charging curve. Experimental results demonstrated that the proposed method achieved high-precision capacity estimation using only a limited amount of battery feature data. In [31], an improved Ant Lion Optimization algorithm and Support Vector Regression (IALO-SVR) were proposed to address the low estimation accuracy of traditional methods. Battery charge and discharge data were analyzed geometrically, and four health features highly correlated with SoH decline were selected as inputs for the SVR model. The IALO algorithm was used to optimize the kernel parameters of SVR, resulting in an accurate SoH estimation model. The method was verified on NASA battery dataset under different working conditions, and compared with ALO-SVR and SVR. The experimental results demonstrated high estimation accuracy, robustness, and stable estimation error within 2%.

In [32], a fusion model based on the Autoregressive Moving Average (ARMA) model and Elman neural network (NN) was proposed for accurate prediction of the state of health (SoH) of lithium-ion batteries. Grey relation analysis and the entropy weight method were employed to analyze battery features [33]. A Long Short-Term Memory (LSTM) model was established to estimate the SoH of batteries. A transfer learning-based method was proposed to monitor the state of health (SoH) of batteries [34]. A novel data processing method based on maximum mean discrepancy was utilized to eliminate redundant information and minimize the difference between different data distributions. Mutual information was used to show that the correlation between processed data was not decreased. The results indicated that transfer learning played an important role in improving the estimation accuracy of battery SoH. In [35], a computationally efficient Revised Lorentzian-based Voltage Capacity (RL-VC) model was used to accurately capture the voltage plateaus of Lithium-Ion Batteries (LIBs), which reflected the material-level phase transition phenomenon. New features of interest (FOIs) were extracted by fitting the RL-VC model to data collected from the constant-current charging process. Correlation analysis was performed on the captured FOIs, and linear models were calibrated to estimate battery SoH. The proposed method showed high accuracy for battery SoH estimation and robust performance against the initial aging status and practical cycling condition. A novel online synthesis method based on the fusion of partial incremental capacity and Artificial Neural Network (ANN) was proposed to estimate SoH and Remaining Useful Life (RUL) under constant current discharge [36]. Advanced filter methods were applied to smooth the initial incremental capacity curves, and strong correlation feature values were extracted using correlation analysis.

The majority of the studies have limited consideration of real-world implementation challenges for electric vehicles. Factors such as data acquisition costs, real-time processing requirements, and practical feasibility need to be addressed to ensure the proposed methods can be readily applied in real-world applications. For example, many studies have used the data from discharge cycle for estimation of SoH where the discharge current was constant. However, this is not a practical case for EVs as the discharge current greatly varies during operation. Several features were used to make the prediction which may by costly or time consuming to obtain during real-world applications. Also, recent state-of-the-art boosting ensemble methods such as LightGBM, Catboost have not been explored so far as of our knowledge. Multivariate feature selection technique such as Sequential Feature Selection is also not used in any of the studies.

CHAPTER 3

BACKGROUND THEORY

3.1 Machine Learning Models

In this section, we discuss the machine learning models used in this study, namely Random Forest, XGBoost, LightGBM, and CatBoost. We will present their respective key concepts and reasons for using each model.

3.1.1 Random Forest

Random Forest [37] is an ensemble learning method that constructs multiple decision trees during training and outputs the mean prediction of the individual trees for regression tasks. The main idea behind Random Forest is to combine the predictions of several decision trees to reduce overfitting and improve generalization.

Given a dataset $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, the Random Forest algorithm builds K decision trees $\{T_1, \ldots, T_K\}$, where each tree T_k is trained on a bootstrap sample drawn from the original dataset. The final prediction is the average of the predictions from all decision trees:

$$\hat{y} = \frac{1}{K} \sum_{k=1}^{K} T_k(x) \tag{3.1}$$

Random Forest is popular due to its ease of use, robustness to overfitting, and ability to handle a large number of features and complex interactions. It can also provide feature importance scores and handle missing values effectively.

3.1.2 XGBoost

XGBoost (eXtreme Gradient Boosting) [38] is an efficient and scalable implementation of the Gradient Boosting Machines algorithm. It is an ensemble learning method that builds multiple weak learners (decision trees) sequentially, with each new tree aiming to correct the errors made by the previous trees.

Given a dataset $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, the XGBoost algorithm constructs an additive model of the form:

$$\hat{y}(x) = \sum_{k=1}^{K} f_k(x) \tag{3.2}$$

where $f_k(x)$ are decision trees and K is the number of boosting rounds. The trees are added sequentially to minimize a given objective function, which consists of a loss function and a regularization term.

XGBoost is well-suited for handling large and high-dimensional datasets, as it is designed for computational efficiency and parallelization. It has built-in regularization and can automatically handle missing values, making it a popular choice for many machine learning competitions and real-world applications.

3.1.3 LightGBM

LightGBM (Light Gradient Boosting Machine) [39] is another efficient implementation of the Gradient Boosting Machines algorithm. It is designed to be more computationally efficient than XGBoost by utilizing a novel tree-growing technique called Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB).

LightGBM builds trees in a leaf-wise manner, instead of the traditional level-wise manner. This allows the model to achieve better accuracy by focusing on regions with larger gradients, but it may also result in overfitting for small datasets.

Similar to XGBoost, LightGBM is suitable for large and high-dimensional datasets. It is particularly well-suited for cases where the dataset is too large to fit in memory, as it can efficiently handle sparse data and categorical features without requiring one-hot encoding.

3.1.4 CatBoost

CatBoost (Categorical Boosting) [40] is a gradient boosting algorithm specifically designed to handle categorical features effectively. It uses a combination of one-hot encoding and mean encoding to handle categorical features, along with a unique ordered boosting technique to reduce overfitting.

Similar to XGBoost and LightGBM, CatBoost builds an additive model of the form:

$$\hat{y}(x) = \sum_{k=1}^{K} f_k(x) \tag{3.3}$$

where $f_k(x)$ are decision trees and K is the number of boosting rounds. The trees are added sequentially to minimize a given objective function, which consists of a loss function and a regularization term.

CatBoost's primary advantage is its ability to handle categorical features effectively without requiring extensive preprocessing. It also provides built-in support for handling missing values and offers various regularization techniques to reduce overfitting. The ordered boosting technique used by CatBoost introduces a random permutation of the training data at each boosting step, allowing the algorithm to reduce overfitting while maintaining the high accuracy of gradient boosting.

In conclusion, each of these machine learning models has its advantages and limitations. Random Forest is an excellent choice for its simplicity and robustness, while XGBoost, LightGBM, and CatBoost offer more advanced boosting techniques for improved performance. The choice of which model to use depends on the specific problem, the characteristics of the data, and the desired trade-offs between computational efficiency, accuracy, and handling of categorical features.

CHAPTER 4

METHODOLOGY

4.1 Data Collection

The data used in this study has been obtained from two sources: NASA Prognostics Center of Excellence (PCoE) [41] and University of Maryland Center for Advanced Life Cycle Engineering (CALCE) [42]. Both sources provide valuable datasets related to battery degradation, which are crucial for the development and evaluation of SoH estimation models. The datasets contain information about cycles, temperature, voltage, current and capacity, among other variables for battery charge and discharge, enabling the investigation of various factors that affect battery performance and degradation.

Dataset-1 comprises data from four lithium-ion battery cells, including their charge and discharge cycles, capacity fade, and impedance growth under different operational conditions. This dataset has been widely used in the research community to develop prognostic models for lithium-ion batteries. The dataset can be accessed from the NASA PCoE website [1].

Dataset-2 contains information on four lithium-ion battery cells. The dataset is available for download from the CALCE website. However, the original dataset is way too complex, requires significant preprocessing [43] and not all information are needed for our study. So, we used a simplified version of that from this link [2].

By leveraging the data from both sources, this study aims to create a comprehensive and robust SoH estimation model. The description of both datasets is given in Table 4.1.

Table 4.1: Summary of the datasets used in this study

Description	Dataset-1	Dataset-2
Cell (form size chemistry)	18650 2 Ah	prismatic 1.1 Ah LCO
No. of cells	4	4

[1]Dataset-1: https://www.nasa.gov/content/prognostics-center-of-excellence-data-set-repository
[2]Dataset-2: https://github.com/konkon3249/BatteryLifePrediction/tree/master

4.2 Data Analysis

4.2.1 Dataset-1

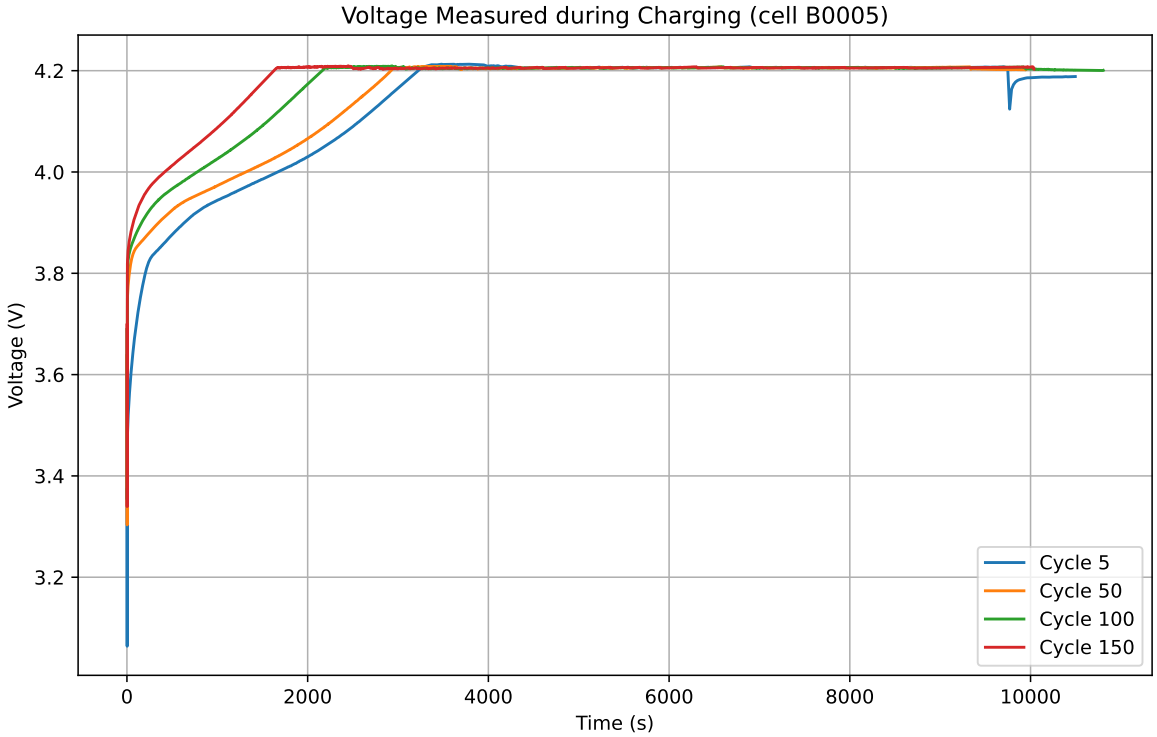


Figure 4.1: Voltage curve during charging for different cycles (cell B0005)

From figure 4.1, it can be observed that as the battery is charged and discharged, the time it requires to reach certain voltage levels changes. This information can be used to estimate battery SoH using ML techniques.

4.2.2 Dataset-2

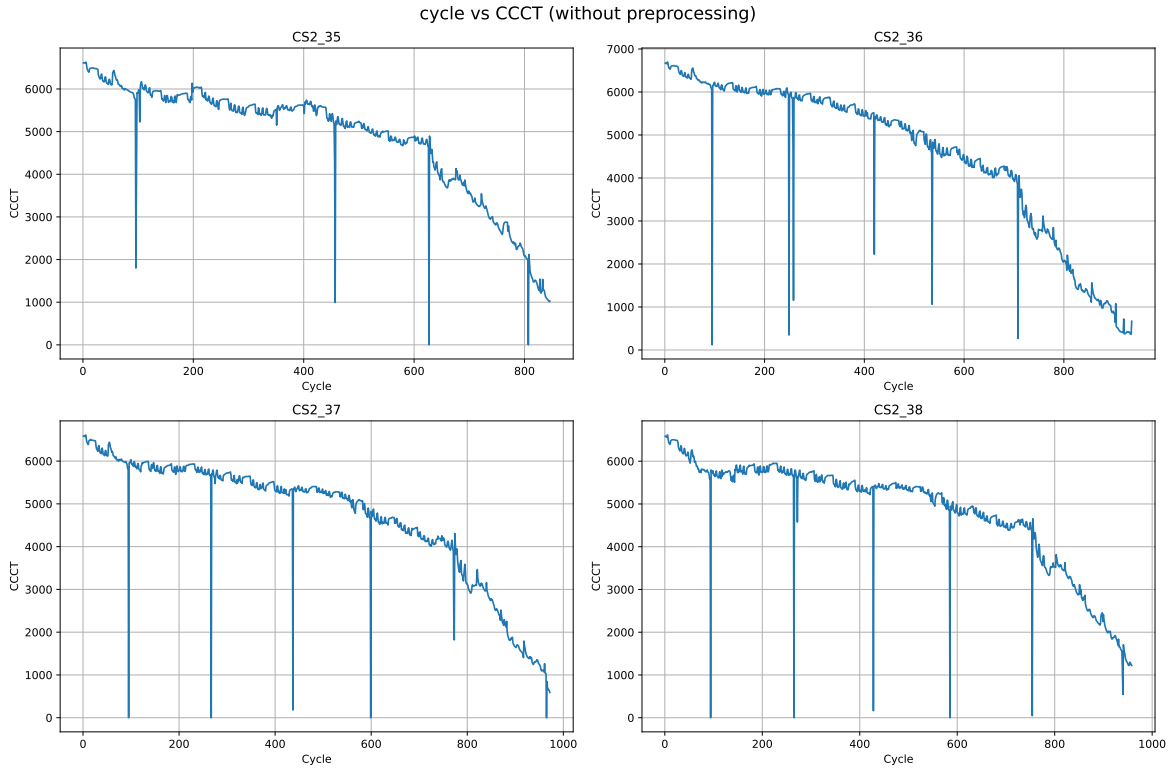


Figure 4.2: Cycle vs CCCT for different cells

Figure 4.2 shows the constant current charge time vs cycle for four cells in dataset-2. The presence of outliers can be easily observed from here. These outliers can significantly hamper the prediction of models.

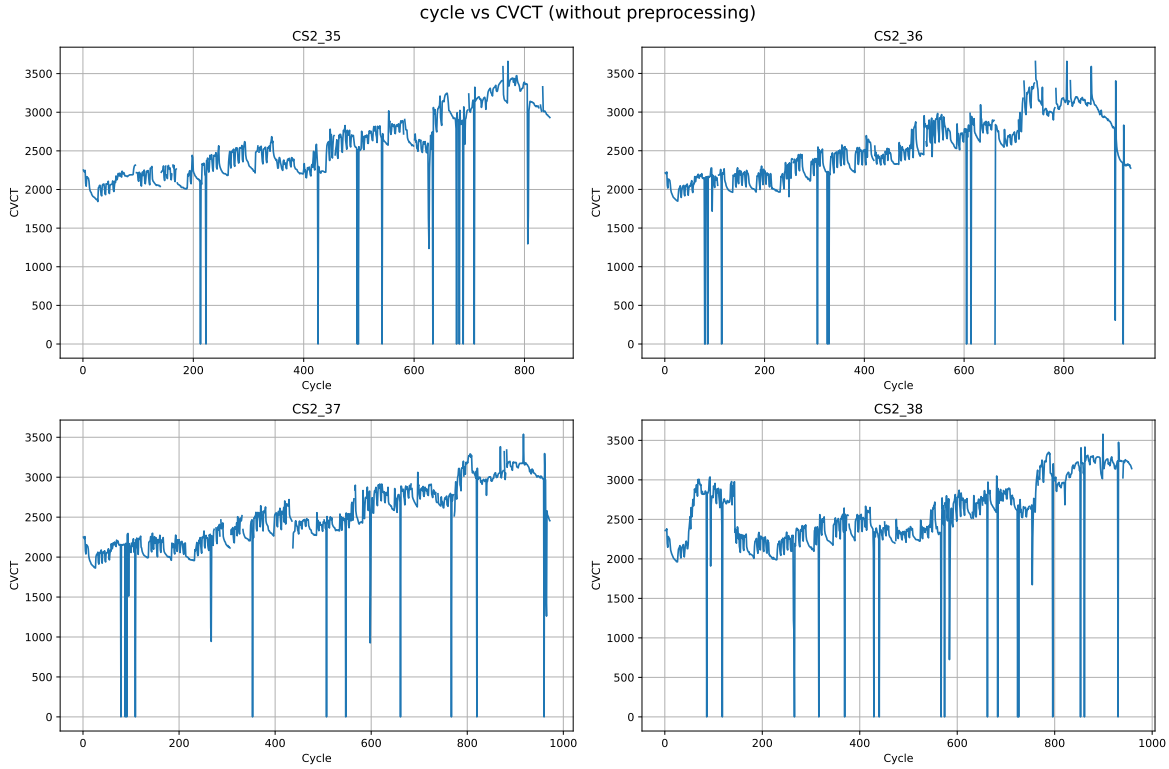


Figure 4.3: Cycle vs CVCT for different cells

Similarly, figure 4.3 shows the presence of outliers for constant current charge time. Here, the no. number of outliers are even higher. The presence of this type of outliers is natural and in order to get optimal performance, they should be handled properly.

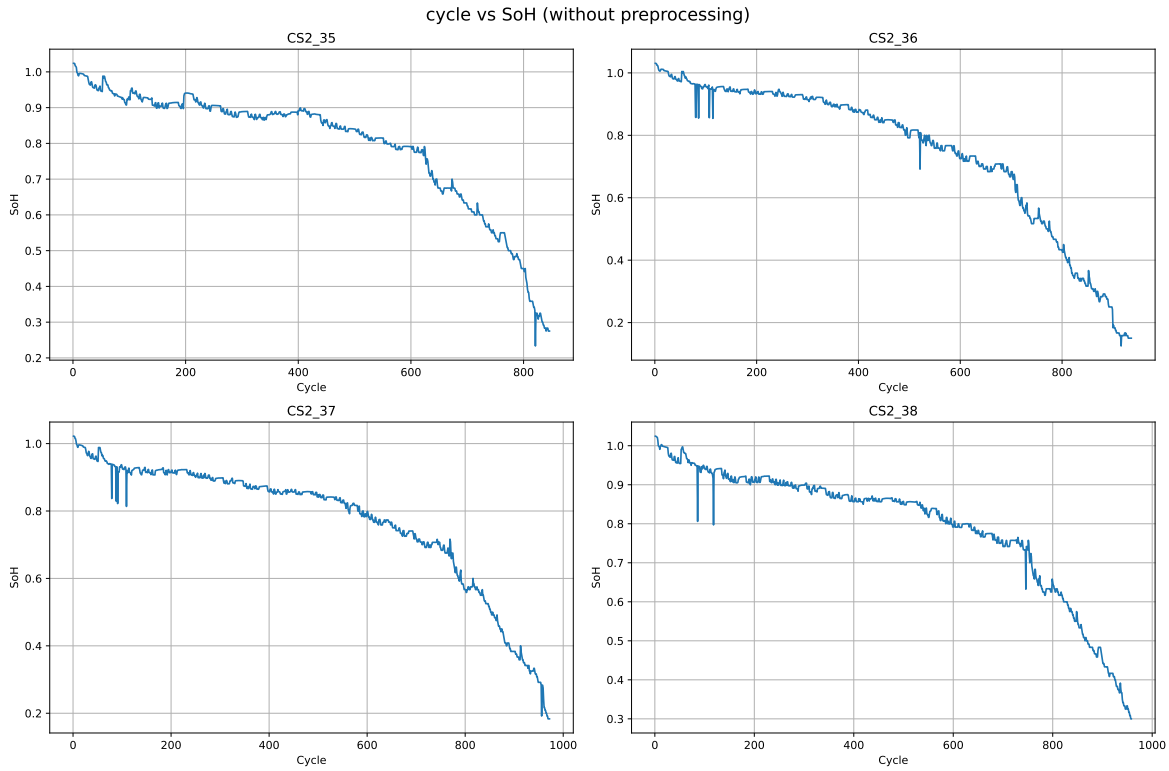


Figure 4.4: Cycle vs SoH for different cells

Figure 4.4 shows SoH for different cycles. Here, the no. of outliers are way less that previous cases. However, for three out of four cells, a different pattern can be seen at early cycles.

To handle with these types of outliers, data preprocessing was done which is described in details in Section 4.3.

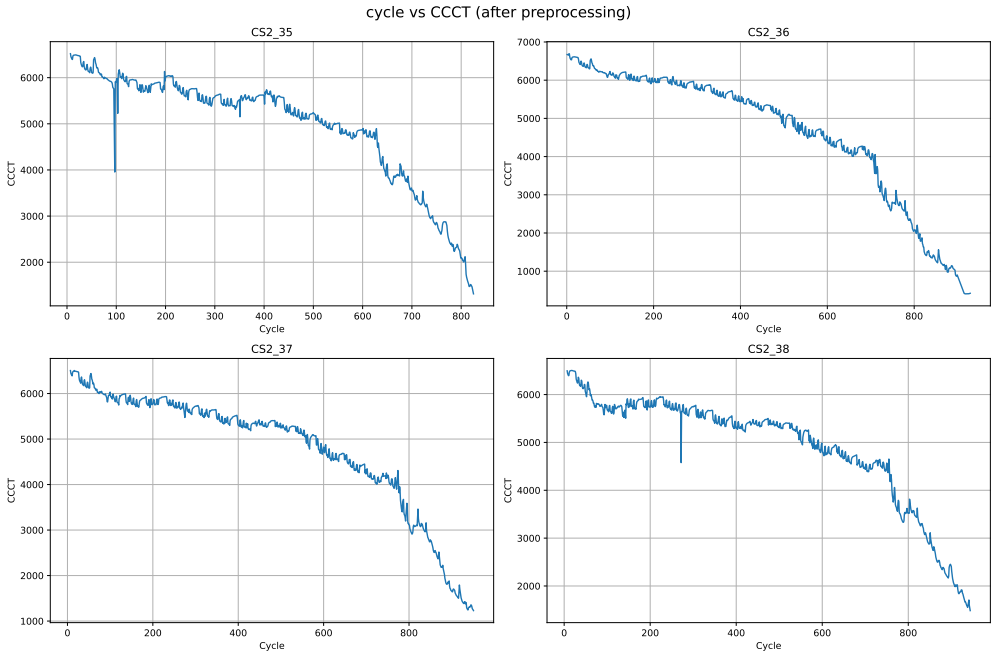


Figure 4.5: Cycle vs CCCT for different cells after data preprocessing

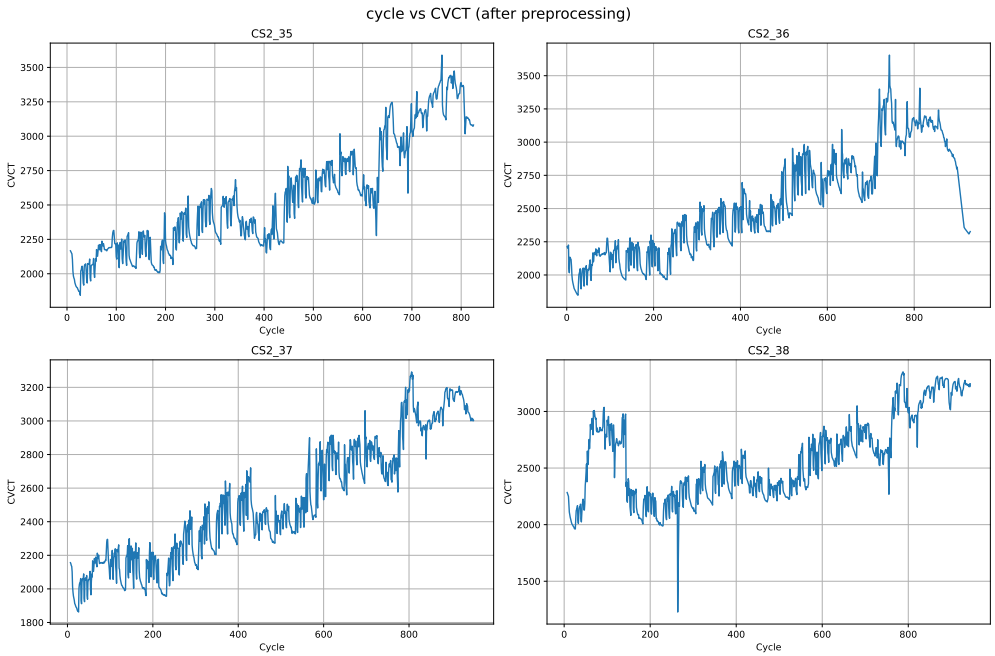


Figure 4.6: Cycle vs CVCT processed for different cells after data preprocessing

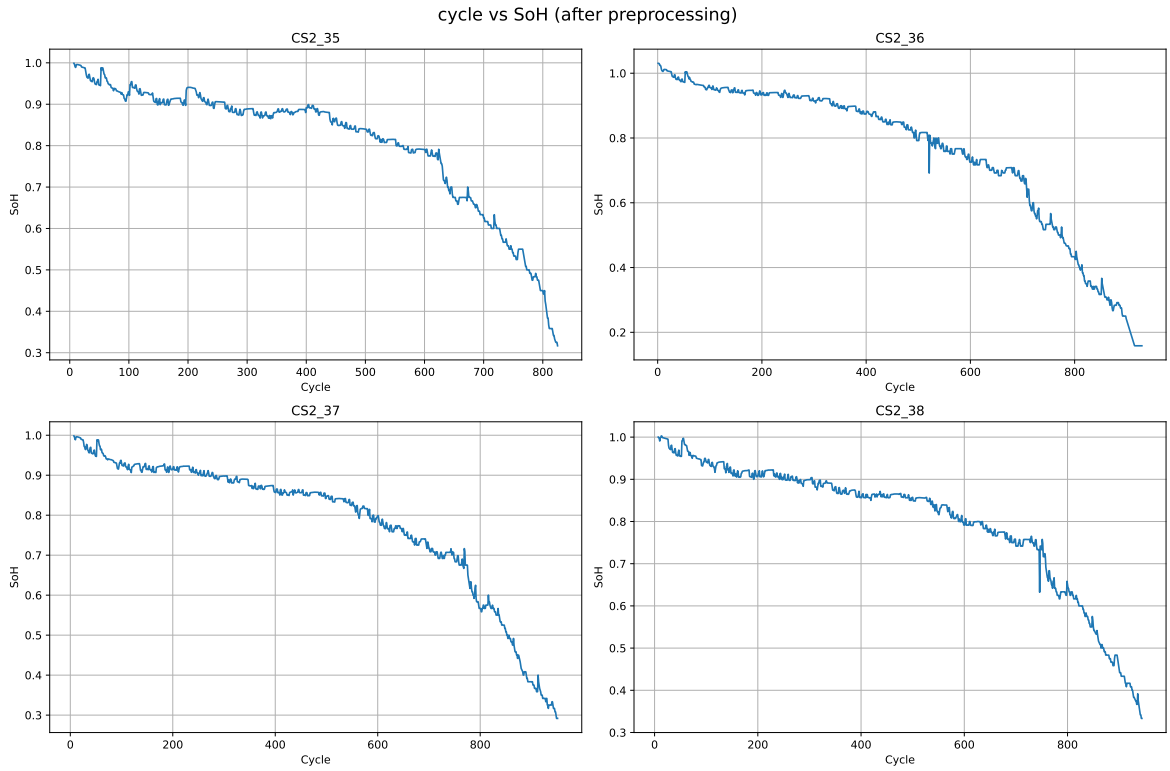


Figure 4.7: Cycle vs SoH processed for different cells after data preprocessing

Figures 4.5, 4.6, 4.7 shows the CCCT, CVCT and SoH for different cycles after data preprossessing. We can clearly see that after data preprocessing, the large deviations from the pattern is greatly reduced. However, the small deviations are allowed to make it close to the actual scenario.

4.3 Data Preprocessing

Data preprocessing is a critical phase where raw data is transformed into a structured format, apt for further analysis and model building. The following subsections delineate the processes involved in the extraction and preprocessing of the charge cycles from the available datasets.

4.3.1 Data Acquisition

The battery data obtained for dataset-1 was in MATLAB's .mat format. A Python-based approach was adopted to convert these files into a more accessible format, like pandas

DataFrame, to facilitate further analysis and machine learning techniques. This conversion process involved loading the charge and discharge cycles from the .mat files and extracting relevant features for each cycle. The extracted features were then aggregated into a list, which was subsequently converted into a pandas DataFrame.

The data for dataset-2 was already in a .csv format containing pertinent information about charge cycles and capacity, hence requiring no additional preprocessing.

4.3.2 Feature Engineering

Feature engineering is a process that improves the performance of machine learning models by creating new features or modifying existing ones. In this context, relevant features were extracted from the raw data, and new features that might give additional insights into the battery's SoH were created. These features were used as inputs for our machine learning models, enhancing their predictive accuracy and overall performance.

The first step in this process was to compute the SoH for each cycle of each battery cell, which represents the current capacity of the battery relative to its nominal capacity. For both datasets, the SoH was computed for each cycle.

For dataset-1, once the SoH was calculated, the charge cycle data for each cell was further processed. This involved extracting the time at which the voltage values crossed specific thresholds and the time it took for the voltage to drop from its peak to the end of the cycle. This resulted in the creation of five features which are explained below:

- time 3.9v: Time to took for the battery voltage to rise to 3.9V during charging.
- time 4.0v: Time to took for the battery voltage to rise to 4.0V during charging.
- time 4.1v: Time to took for the battery voltage to rise to 4.1V during charging.
- time 4.2v: Constant current charge time.
- time end: Constant voltage charge time.

4.3.3 Data Cleaning

Data cleaning is a fundamental step in preparing the data for model training. This study involved addressing missing values and handling outliers to enhance the quality of the data.

Handling Missing Values

As the no. of samples that contained missing values were relatively small, those samples were removed from further analysis.

Handling Outliers

Outliers, defined as data points that deviate significantly from the majority of the data, can have a detrimental impact on the performance of machine learning models. In our study, we considered outliers based on the positional information of the samples, not based on their distribution.

To tackle this challenge, we employed an unsupervised learning method called Isolation Forest [44] for anomaly detection. This technique identifies and isolates outliers by constructing random decision trees. By measuring the average path length required to isolate a data point, the Isolation Forest algorithm effectively distinguishes outliers as instances that have shorter average path lengths.

4.3.4 Feature Scaling

In many machine learning applications, it is common to scale features to improve the performance of the algorithms. Two popular methods for feature scaling are MinMaxScaler and StandardScaler. MinMaxScaler scales features by transforming them to a specific range, typically between 0 and 1, while StandardScaler standardizes features by centering the mean at 0 and scaling to unit variance.

Initially, we attempted to scale the features using both MinMaxScaler and StandardScaler. However, we observed that the algorithms were making constant predictions for all test cases after scaling. This could be attributed to the loss of information during the scaling process or the potential amplification of noise present in the data. Another possible explanation is that the algorithms might not be sensitive to variations in the scaled features, causing them to predict a constant value for all test cases.

Considering these issues, we decided not to scale the features. Since all the features in our dataset are in the same range (time), the absence of scaling should not negatively impact the performance of our model.

4.4 Feature Selection

Feature selection is a crucial step in the process of building a machine learning model. It helps in reducing the number of input variables, thus improving the model's performance and reducing the computational complexity. In this study, we applied four different feature selection techniques to identify the most important features for estimating SoH. These techniques include F regression-based feature selection, mutual information-based feature selection, sequential feature selection, and recursive feature elimination.

This part was only done for dataset-1. Dataset-2 has only 2 useful features so this types of feature selection was not necessary.

4.4.1 F-Regression-based Feature Selection

F-regression-based feature selection involves selecting features based on their correlation with the target variable using the F-test. The F-test measures the significance of the relationship between each feature and the target variable. Features with higher F-statistics and lower p-values are considered more relevant for prediction. This technique helps reduce dimensionality and improve model performance by focusing on the most informative features. The top 2 features was selected by this method.

4.4.2 Mutual Information-based Feature Selection

Mutual information-based feature selection [45] measures the dependence between the input features and the target variable. Higher mutual information values indicate a stronger relationship between the feature and the target variable. The top 2 features with the highest mutual information scores are selected for further analysis.

4.4.3 Sequential Feature Selection

Sequential feature selection is a greedy search algorithm that aims to find the optimal subset of features by iteratively adding or removing features based on the model's performance. In this study, we used the Sequential Feature Selector (SFS) from the MLxtend library [46], which implements a floating search algorithm. The algorithm starts with the full feature set and iteratively removes features to maximize the model's performance, which is known as Sequential Backward Selection. We used the negative mean squared error as the performance

metric. For cross-validation, all samples from one battery cell was used for testing and all samples from the other three cells was used for training. This was done to ensure that every condition possible is tested properly.

4.4.4 Recursive Feature Elimination

Recursive feature elimination with cross-validation (RFECV) is a method that recursively removes the least important features and builds the model using the remaining features. The process is repeated until the optimal number of features is found. We used the RFECV implementation available in the scikit-learn library [47], with the negative mean squared error as the performance metric and the same cross-validation strategy as 4.4.3.

After applying all the feature selection techniques, the performance with selected features from each method are compared, and the best set of features are chosen for building the models.

4.5 Hyperparameter Optimization

Hyperparameter tuning is a critical step in the development of machine learning models, as it allows for the optimization of model performance by fine-tuning the model's parameters. In our study, we performed hyperparameter tuning for the ML models used in this study using Scikit-learn Optimize library [48]. The library offers optimization algorithms, such as Gaussian Process optimization, which we used in this study (referred to as `gp_minimize`).

4.5.1 Parameter Grids

The search space for each model is given in Table 4.2.

Table 4.2: Search space for hyperparameter optimization

Random Forest			
n_estimators	[10, 1000] (integer)	min_samples_split	$[10^{-5}, 10^{0}]$ (log-uniform)
max_depth	[1, 20] (integer)	min_samples_leaf	$[10^{-5}, 10^{-1}]$ (log-uniform)
XGBoost			
max_depth	[1, 20] (integer)	gamma	$[10^{-5}, 10^{1}]$ (uniform)
learning_rate	$[10^{-5}, 10^{0}]$ (log-uniform)	alpha	$[10^{-5}, 10^{0}]$ (uniform)
subsample	[0.5, 1] (uniform)		
LightGBM			
max_depth	[2, 20] (integer)	reg_alpha	$[10^{-5}, 10^{1}]$ (uniform)
learning_rate	$[10^{-5}, 10^{0}]$ (log-uniform)	reg_lambda	$[10^{-5}, 10^{1}]$ (uniform)
subsample	[0.5, 1] (uniform)		
CatBoost			
depth	[1, 10] (integer)	learning_rate	[0.01, 0.5] (uniform)
l2_leaf_reg	[0.1, 10] (uniform)	subsample	[0.01, 1] (uniform)

4.5.2 Objective Function

We defined an objective function to be minimized during the optimization process. The objective function computes the average of the cross-validation RMSE for a given set of hyperparameters. We used the negative RMSE as the scoring metric to turn the problem into a minimization task.

Gaussian Process Optimization

We employed the Gaussian Process optimization method implemented in the Scikit-Optimize library. We used the `gp_minimize` function with a total of 200 optimization calls.

After the optimization process is completed, the model with the best hyperparameters is saved for future use.

In summary, the hyperparameter optimization process aims to find the best possible combination of hyperparameters for each regression model. By tuning the models' hyperparameters, we can improve their performance and achieve better results in predicting the target variable.

CHAPTER 5

RESULT ANALYSIS

5.1 Evaluation Metrics

In order to evaluate the performance of the models on the two datasets, five commonly used metrics for regression tasks, namely MAE, MSE, RMSE, R-squared (R^2), and MAPE are used in our research. These are explained in detail.

5.1.1 Mean Absolute Error (MAE)

Mean Absolute Error (MAE) is a widely used evaluation metric in regression problems. It measures the average of the absolute differences between the predicted and actual values.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{5.1}$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of samples.

MAE is easy to understand and interpret, as it represents the average error magnitude without considering the direction of the error. However, it is less sensitive to outliers since it does not square the errors.

5.1.2 Mean Squared Error (MSE)

Mean Squared Error (MSE) is another popular evaluation metric in regression tasks. It measures the average of the squared differences between the predicted and actual values.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5.2}$$

MSE is sensitive to outliers as it squares the errors, making larger errors more prominent. It is used when it is crucial to penalize larger errors more heavily than smaller ones.

5.1.3 Root Mean Squared Error (RMSE)

Root Mean Squared Error (RMSE) is the square root of the Mean Squared Error (MSE). It represents the standard deviation of the residuals (prediction errors) and is defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{5.3}$$

RMSE has the same unit as the target variable, making it easier to interpret than MSE. It is sensitive to outliers and is commonly used when larger errors should be penalized more than smaller errors.

5.1.4 R-squared (R^2)

R-squared (R^2), also known as the coefficient of determination, measures the proportion of the variance in the dependent variable that can be explained by the independent variables. It is a popular evaluation metric that ranges from 0 to 1, with higher values indicating a better fit.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5.4}$$

where $\bar{y}$ is the mean of the actual values.

R^2 is widely used because it provides a single value that describes the goodness of fit of a model. However, it has limitations, such as its inability to determine whether the coefficients and predictions are biased.

5.1.5 Mean Absolute Percentage Error (MAPE)

Mean Absolute Percentage Error (MAPE) measures the average percentage error between the predicted and actual values.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100 \tag{5.5}$$

MAPE is often used when it is essential to express the error as a percentage of the actual value. It provides an easily interpretable error metric and is commonly used in forecasting and time series analysis. However, MAPE has some drawbacks, such as being undefined

when $y_i = 0$ and being biased towards underestimating high values.

In conclusion, each of these evaluation metrics has its advantages and limitations. The choice of which metric to use depends on the goals we are trying to achieve and the characteristics of the data. By considering multiple metrics, we can obtain a more comprehensive understanding of a model's performance. In our case, where larger errors can lead to safety issues, MSE and RMSE carry higher weightage.

5.2 Dataset-1

5.2.1 Feature Selection

Correlation & Mutual Information

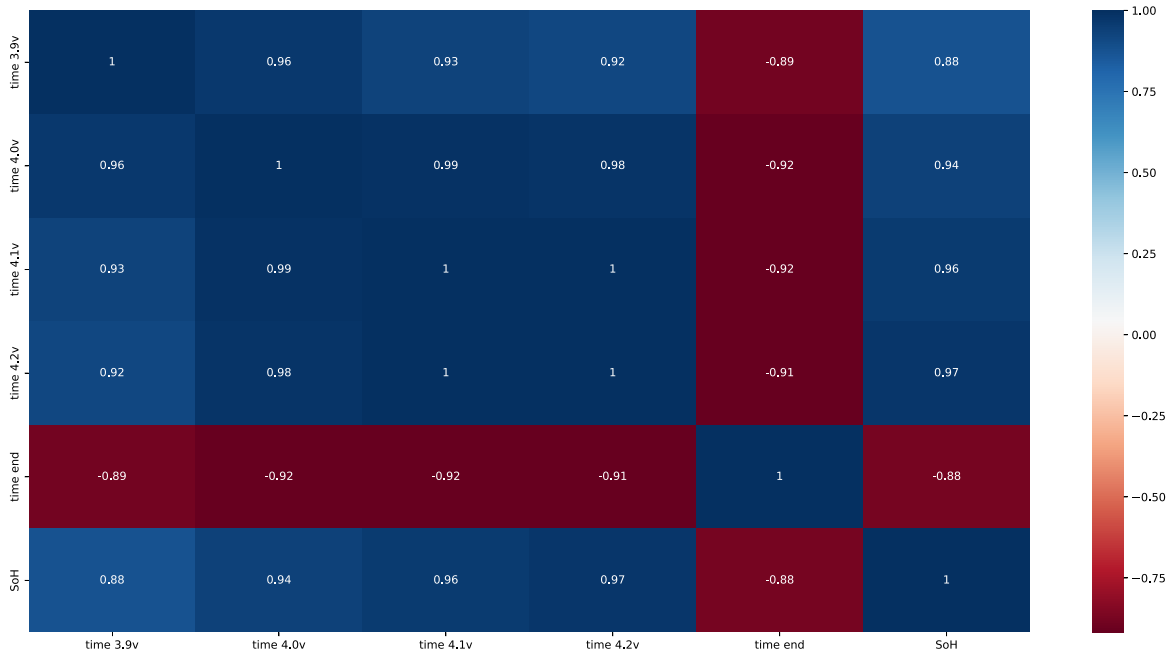
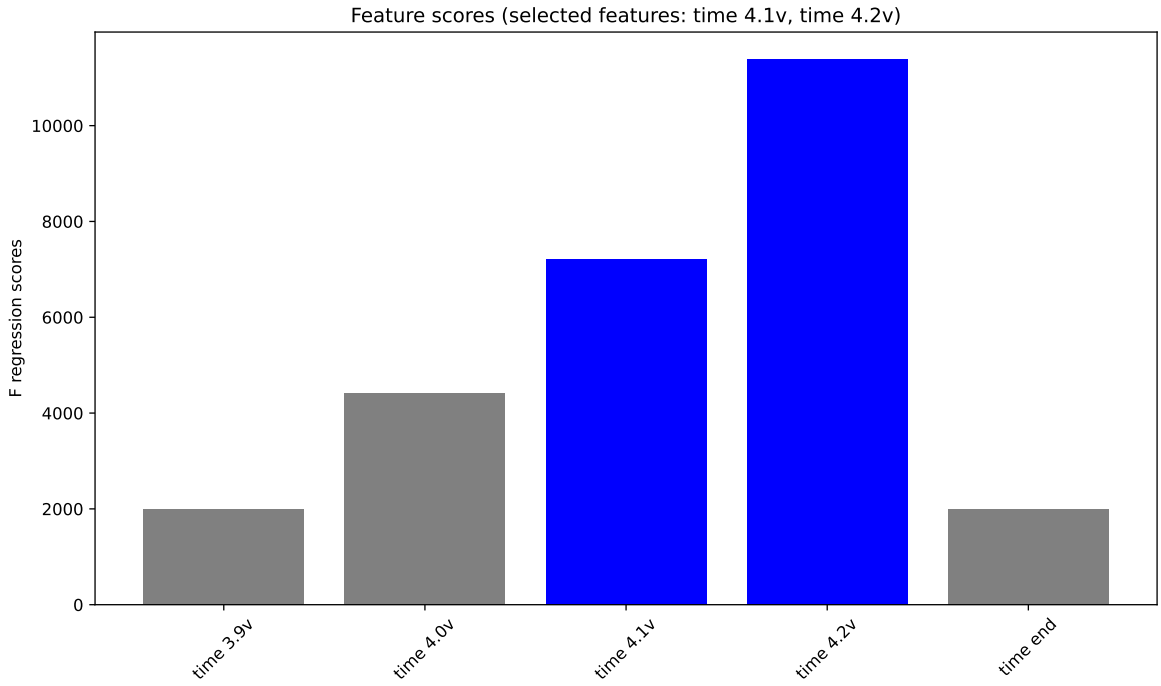


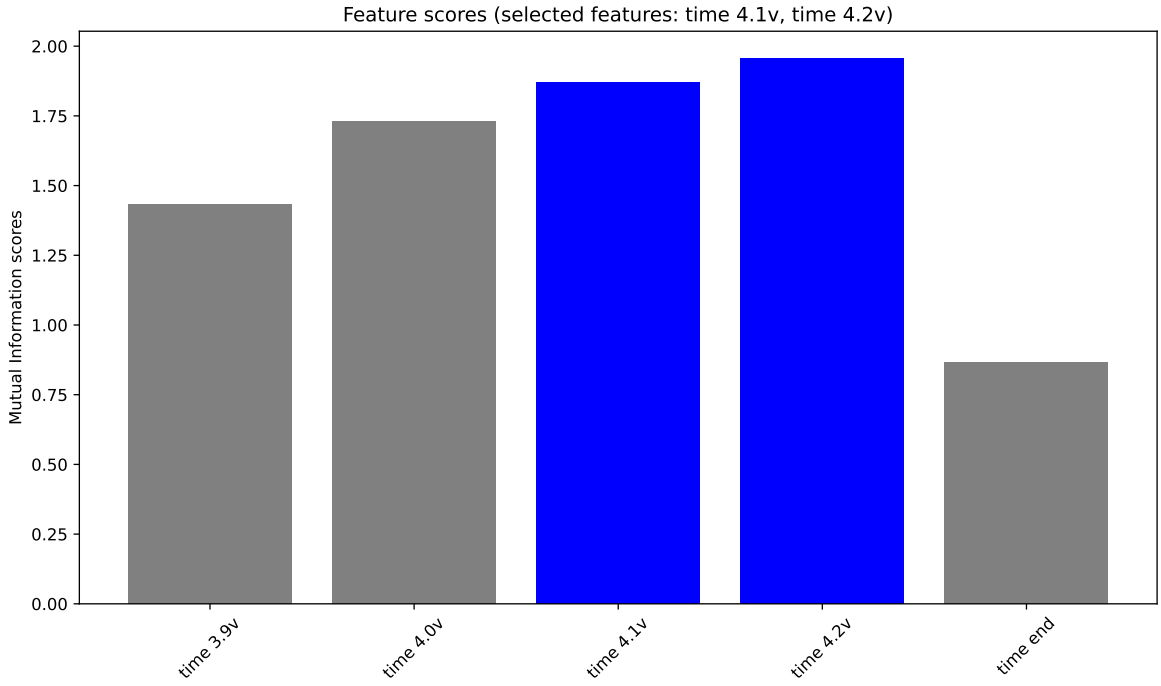
Figure 5.1: Correlation of the extracted features with SoH

Figure 5.1 shows the Pearson correlation between between the selected features and the target. We can observe that the features are highly correlated.

We also utilized both F-regression and mutual information to select the best features. The top 2 features selected based on each method are presented below.



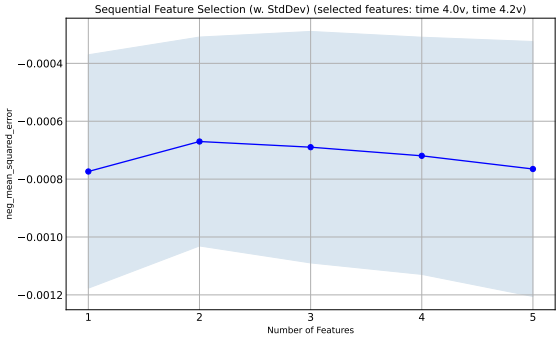
(a) Top 2 features selected by F-regression scores



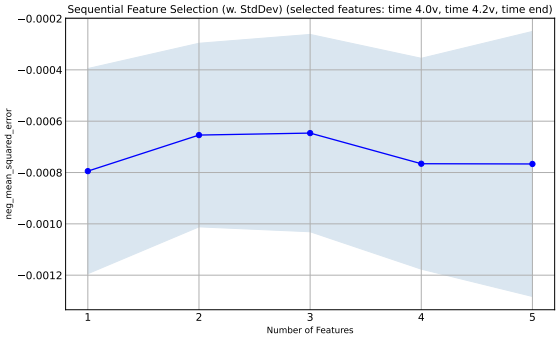
(b) Top 2 selected features using Mutual Info Regression

Figure 5.2: Top features selected via F regression and Mutual Information scores

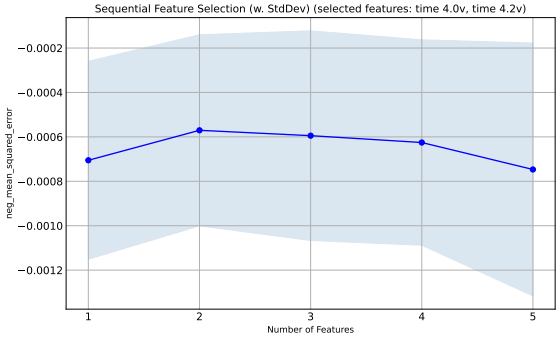
Figure 5.2 presents the selected features based F regression and Mutual Information. In Figure 5.2a, the selection was done based on the F regression value while Figure 5.2b presents the selected features based on Mutual Information.



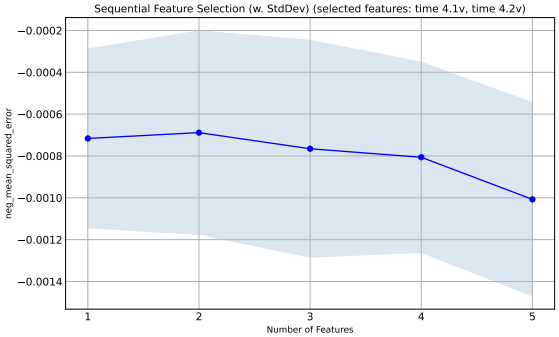
(a) Best features selected for Random Forest



(b) Best features selected for XGBoost



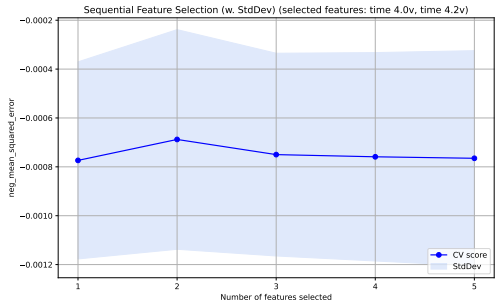
(c) Best features selected for LightGBM



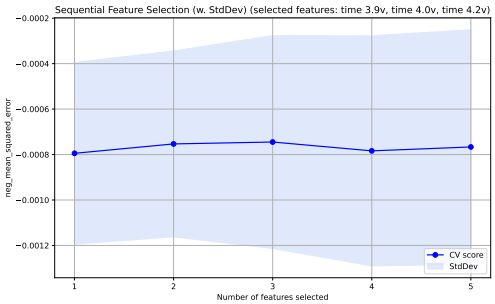
(d) Best features selected for CatBoost

Figure 5.3: Comparison of optimal features selected using Sequential Feature Selection for various models

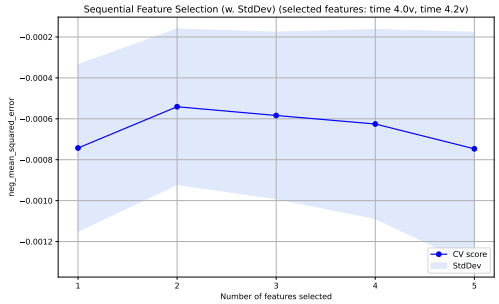
Figure 5.3 illustrates the relationship between the number of best-selected features and the corresponding MSE values for different algorithms. It is evident that the lowest error was achieved when utilizing the features *time 4.2v* and *time 4.0v* for most of the models.



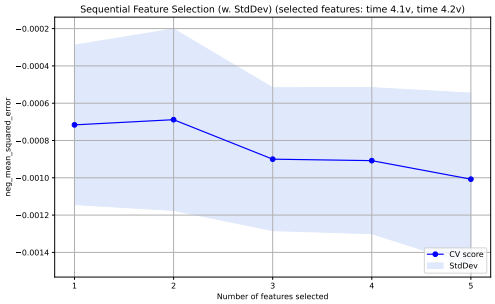
(a) Best features selected for Random Forest



(b) Best features selected for XGBoost



(c) Best features selected for LightGBM



(d) Best features selected for CatBoost

Figure 5.4: Comparison of optimal features selected using RFECV for various models

Figure 5.4 illustrates the relationship between the number of best-selected features and the corresponding MSE loss for different algorithms using Recursive Feature Elimination. It is noteworthy that different models achieved the best results using different sets of features; however, *time 4.2v* and *time 4.0v* remained the most significant features across the models.

Table 5.1 shows the results for different feature selection techniques for RF model across a range of performance metrics. Similarly, Tables 5.2, 5.3, and 5.4 present the results for the XGBoost, LightGBM, and CatBoost models, respectively.

Table 5.1: Comparison of Random Forest model performance for different feature selection methods on dataset-1

	mse	rmse	r2	mae	mape	time
base	0.000 765	0.026 500	0.914 758	0.021 294	0.028 418	3.585 527
f_reg	0.000 676	0.024 422	0.927 370	0.019 376	0.025 775	1.472 194
mi	0.000 676	0.024 422	0.927 370	0.019 376	0.025 775	1.511 047
sfs	0.000 670	0.024 956	0.924 130	0.019 971	0.026 642	0.327 196
rfe	0.000 670	0.024 956	0.924 130	0.019 971	0.026 642	0.322 530

Table 5.2: Comparison of XGBoost model performance for different feature selection methods on dataset-1

	mse	rmse	r2	mae	mape	time
base	0.000 767	0.026 212	0.917 015	0.021 188	0.028 062	0.086 525
f_reg	0.000 660	0.024 014	0.929 970	0.018 935	0.025 195	0.069 518
mi	0.000 660	0.024 014	0.929 970	0.018 935	0.025 195	0.065 990
sfs	0.000 646	0.024 432	0.927 869	0.019 730	0.026 125	0.057 518
rfe	0.000 681	0.025 012	0.924 575	0.019 994	0.026 452	0.046 994

Table 5.3: Comparison of LightGBM model performance for different feature selection methods on dataset-1

	mse	rmse	r2	mae	mape	time
base	0.000 747	0.025 397	0.921 391	0.020 735	0.027 357	0.203 268
f_reg	0.000 678	0.023 804	0.930 023	0.018 747	0.024 837	0.220 207
mi	0.000 678	0.023 804	0.930 023	0.018 747	0.024 837	0.206 041
sfs	0.000 570	0.022 453	0.938 683	0.018 696	0.024 687	0.031 021
rfe	0.000 570	0.022 453	0.938 683	0.018 696	0.024 687	0.033 515

Table 5.4: Comparison of Catboost model performance for different feature selection methods on dataset-1

	mse	rmse	r2	mae	mape	time
base	0.001 007	0.030 781	0.878 782	0.024 639	0.033 006	3.517 787
f_reg	0.000 688	0.024 556	0.926 451	0.019 671	0.026 111	2.327 252
mi	0.000 688	0.024 556	0.926 451	0.019 671	0.026 111	2.393 319
sfs	0.000 688	0.024 556	0.926 451	0.019 671	0.026 111	1.674 881
rfe	0.000 688	0.024 556	0.926 451	0.019 671	0.026 111	1.498 810

We can observe that best RMSE loss was obtained by LightGBM model using *time 4.0v* and *time 4.2v* as features.

Figure 5.5 shows the estimated SoH for the four cells in dataset-1 using the features selected from Sequential Feature Selection for the Random Forest model. The estimated values are almost equal to the actual values for all cells except cell B0006.

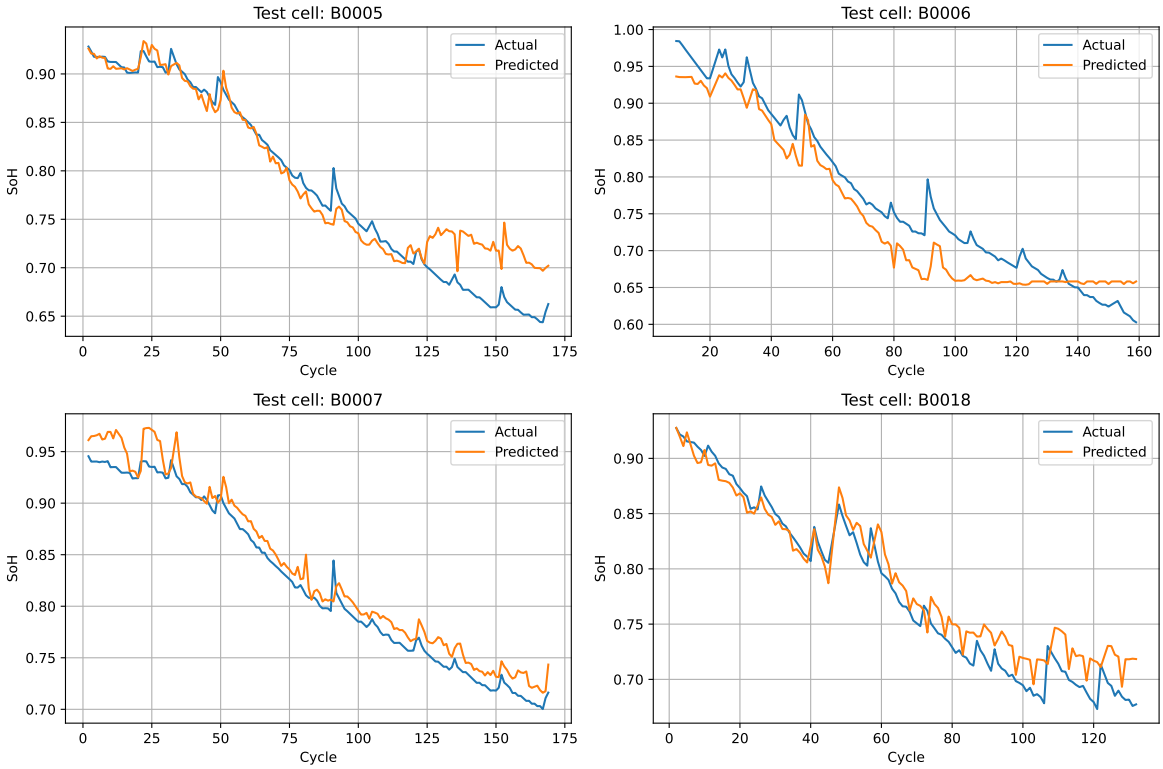


Figure 5.5: SoH estimation using Random Forest on dataset-1

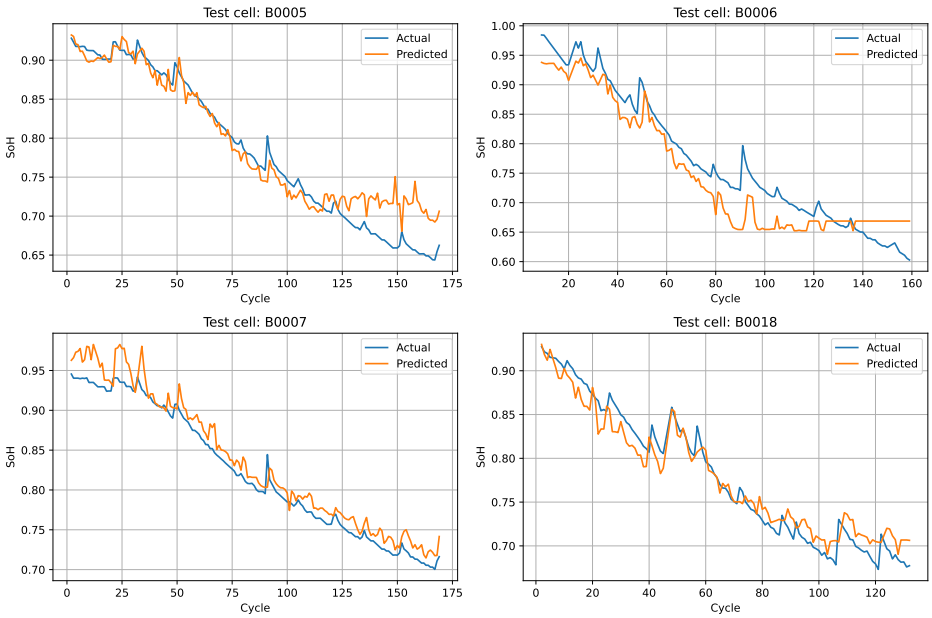


Figure 5.6: SoH estimation using XGBoost on dataset-1

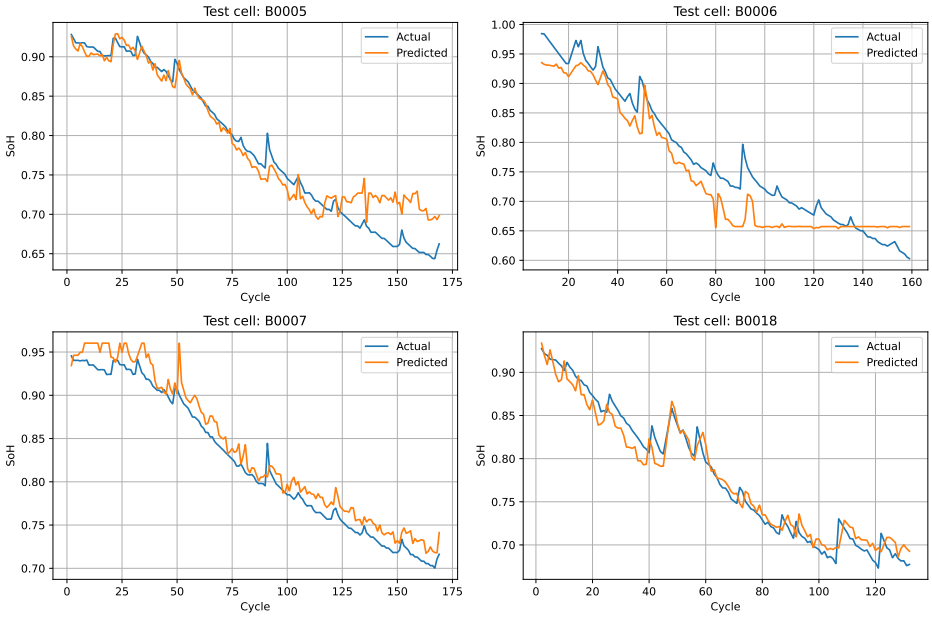


Figure 5.7: SoH estimation using LightGBM on dataset-1.

Figures 5.6, 5.7, and 5.8 show the estimated SoH for the four cells in dataset-1 using the features selected from Sequential Feature Selection for XGBoost, LightGBM, and Catboost,

respectively. One similar pattern was found: all models struggled to accurately predict the SoH for the last few cycles of cell B0005 and B0006.

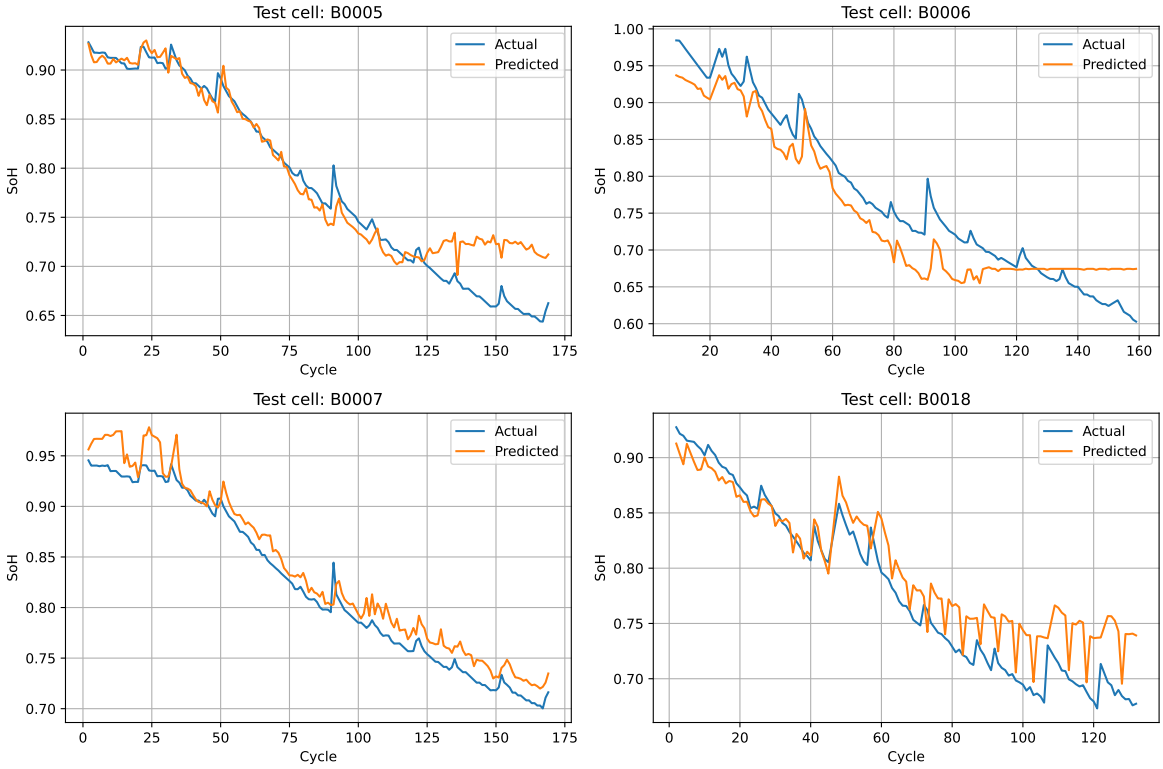


Figure 5.8: SoH estimation using Catboost on dataset-1.

5.2.2 Hyperparameter Optimization

Tables 5.5, 5.6, 5.7 and 5.8 show the performance in terms of the accuracy, loss and time taken after hyperparameter tuning.

Table 5.5: Comparison of tuned Random Forest model performance metrics for different feature selection methods on dataset-1

	mse	rmse	r2	mae	mape	time
base	0.001 024	0.030 867	0.882 173	0.025 671	0.034 013	3.528 839
f_reg	0.001 030	0.030 963	0.881 492	0.025 733	0.034 108	1.368 844
mi	0.001 030	0.030 963	0.881 492	0.025 733	0.034 108	1.339 299
sfs	0.001 010	0.030 656	0.883 656	0.025 494	0.033 775	0.136 528
rfe	0.001 010	0.030 656	0.883 656	0.025 494	0.033 775	0.116 181

Table 5.6: Comparison of tuned XGBoost model performance for different feature selection methods on dataset-1

	mse	rmse	r2	mae	mape	time
base	0.000 745	0.024 500	0.924 712	0.020 282	0.026 656	2.285 072
f_reg	0.000 689	0.023 390	0.930 671	0.019 180	0.025 303	0.867 502
mi	0.000 689	0.023 390	0.930 671	0.019 180	0.025 303	0.870 916
sfs	0.000 652	0.023 274	0.932 768	0.019 562	0.025 695	0.068 005
rfe	0.000 673	0.023 568	0.930 969	0.019 691	0.025 865	0.072 519

Table 5.7: Comparison of tuned LightGBM model performance for different feature selection methods on dataset-1

	mse	rmse	r2	mae	mape	time
base	0.000 757	0.025 951	0.918 561	0.020 939	0.027 535	0.217 339
f_reg	0.000 645	0.023 382	0.932 946	0.018 073	0.023 912	0.205 944
mi	0.000 645	0.023 382	0.932 946	0.018 073	0.023 912	0.199 716
sfs	0.000 549	0.022 252	0.939 232	0.018 465	0.024 309	0.032 102
rfe	0.000 549	0.022 252	0.939 232	0.018 465	0.024 309	0.030 571

Table 5.8: Comparison of tuned Catboost model performance for different feature selection methods on dataset-1

	mse	rmse	r2	mae	mape	time
base	0.001 079	0.031 957	0.863 965	0.025 126	0.033 658	8.053 784
f_reg	0.000 658	0.024 054	0.929 654	0.019 249	0.025 480	3.439 602
mi	0.000 658	0.024 054	0.929 654	0.019 249	0.025 480	3.478 509
sfs	0.000 658	0.024 054	0.929 654	0.019 249	0.025 480	3.004 984
rfe	0.000 658	0.024 054	0.929 654	0.019 249	0.025 480	2.887 118

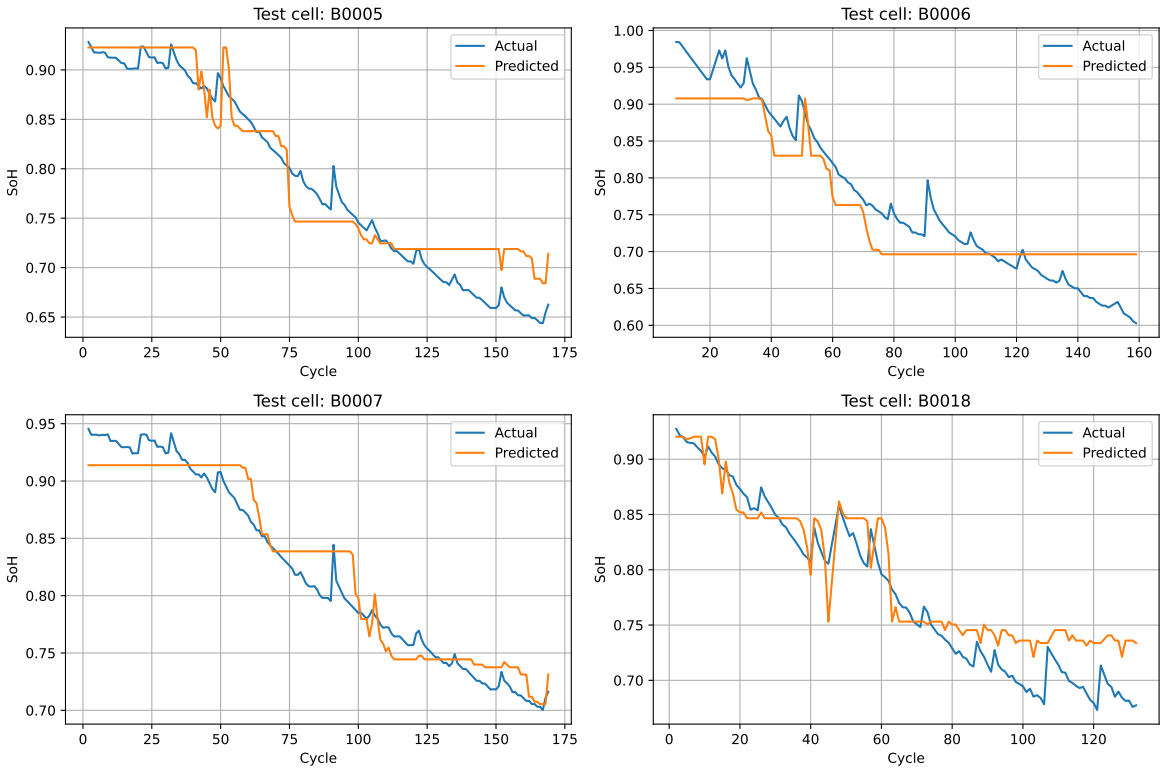


Figure 5.9: SoH estimation using Random Forest on dataset-1 after hyperparameter optimization

Figure 5.9 shows the estimated SoH after hyperparameter optimization on Random Forest model. We can see the prediction quality degraded compared to the default model.

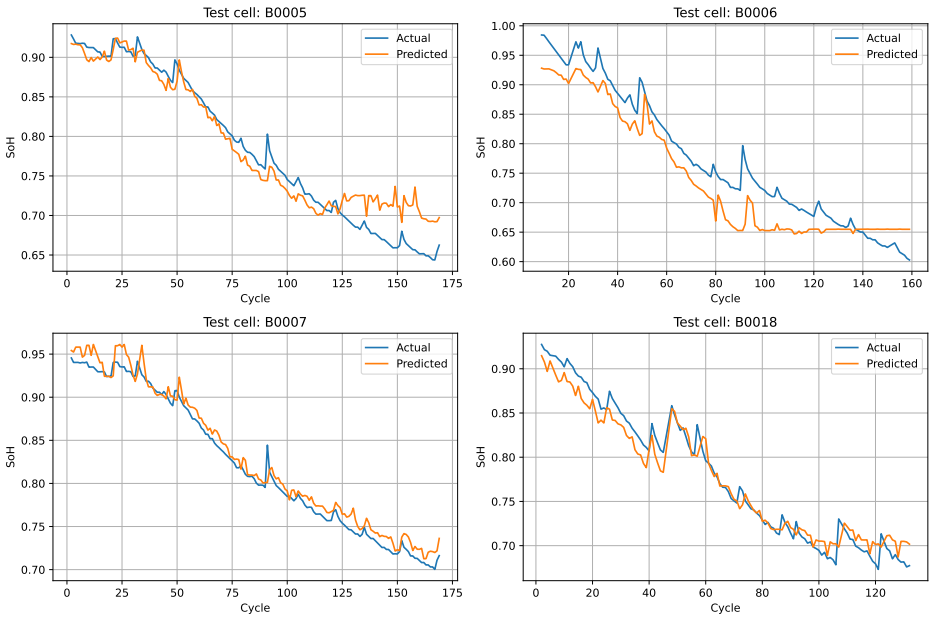


Figure 5.10: SoH estimation using XGBoost on dataset-1 after hyperparameter optimization

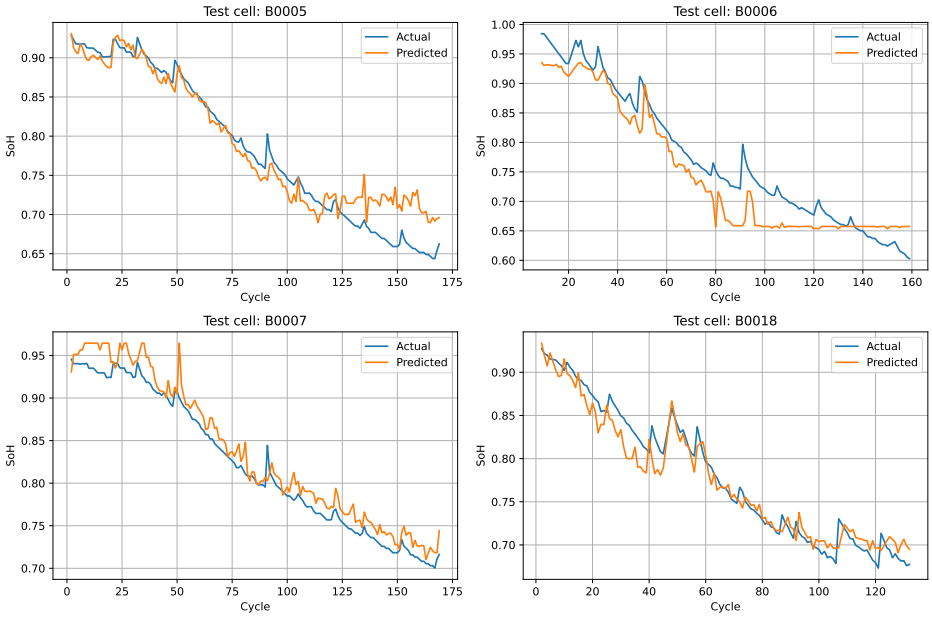


Figure 5.11: SoH estimation using LightGBM on dataset-1 after hyperparameter optimization

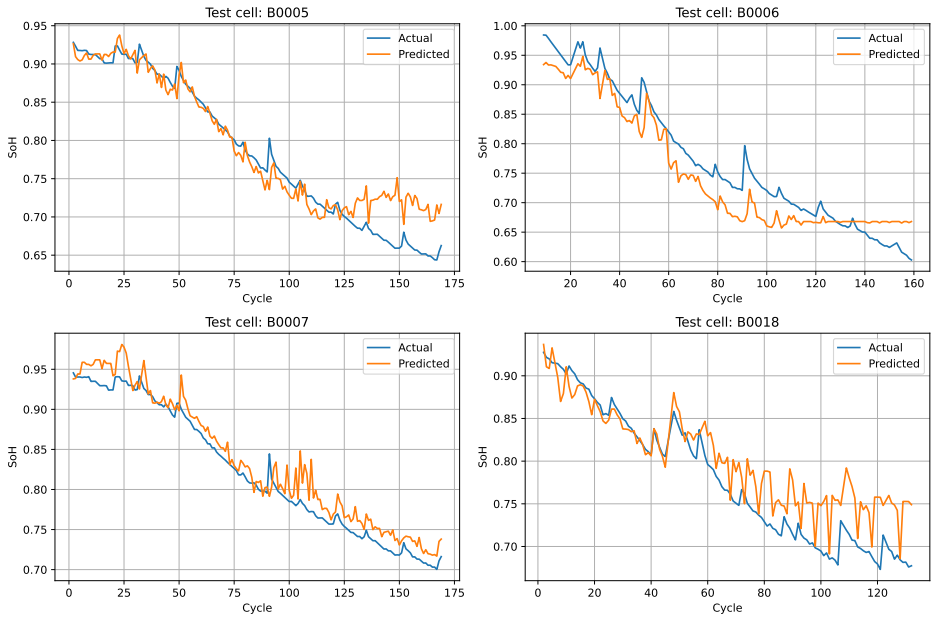


Figure 5.12: SoH estimation using Catboost on dataset-1 after hyperparameter optimization

Figures 5.10, 5.11, and 5.12 present the estimated SoH for the four cells in dataset-1 using XGBoost, LightGBM, and Catboost after hyperparameter optimization. We observe that, in general, the application of hyperparameter optimization had little to no impact on improving the performance of these models.

5.3 Dataset-2

Table 5.9: Comparison of Random Forest model performance for different feature selection methods on dataset-2

	mse	rmse	r2	mae	mape	time
base	0.000 241	0.015 416	0.991 392	0.009 856	0.015 647	2.800 895
CCCT	0.000 376	0.019 336	0.986 247	0.013 125	0.020 050	1.218 569
CVCT	0.011 829	0.106 928	0.590 483	0.067 454	0.116 884	1.302 974

Table 5.10: Comparison of XGBoost model performance metrics for different feature selection methods on dataset-2

	mse	rmse	r2	mae	mape	time
base	0.000 260	0.015 935	0.990 953	0.009 766	0.015 841	0.182 045
CCCT	0.000 329	0.017 973	0.987 873	0.011 922	0.018 383	0.128 517
CVCT	0.010 965	0.102 756	0.621 151	0.062 854	0.111 004	0.127 522

Table 5.11: Comparison of LightGBM model performance for different feature selection methods on dataset-2

	mse	rmse	r2	mae	mape	time
base	0.000 243	0.015 363	0.991 618	0.009 439	0.015 495	3.271 407
CCCT	0.000 300	0.017 056	0.989 518	0.011 361	0.018 129	1.249 964
CVCT	0.008 982	0.092 051	0.696 449	0.058 669	0.104 801	1.201 086

Table 5.12: Comparison of Catboost model performance for different feature selection methods on dataset-2

	mse	rmse	r2	mae	mape	time
base	0.000 527	0.020 536	0.984 915	0.010 250	0.018 990	2.281 120
CCCT	0.000 290	0.016 807	0.989 749	0.011 175	0.017 746	1.989 980
CVCT	0.008 929	0.091 865	0.698 352	0.058 633	0.104 549	2.004 307

For dataset-2, we evaluated three different sets of features: Constant Current Charge Time (CCCT), Constant Voltage Charge Time (CVCT), and a combination of both (referred to as 'base').

Table 5.9 presents the results for the Random Forest model. The model which includes both CCCT and CVCT as features performed well in terms of all metrics. However, the models using only CCCT or CVCT as features yielded lower R^2 scores. Additionally, these models took less time to train compared to the model using both features.

Table 5.10 shows the results for the XGBoost model. Here, all feature sets achieved high R^2 scores, indicating strong predictive performance. The models using only CCCT or CVCT

features took less time to train than the one using both features.

Table 5.11 outlines the performance of the LightGBM model. All feature sets had high R^2 scores. However, the model using CVCT features as a single predictor yielded a lower score than the other two methods. Like the previous case, LightGBM was the best performing algorithm among the four.

Table 5.12 presents the results for the Catboost model. Here, all feature sets exhibited strong predictive performance, as indicated by high R^2 scores. In terms of execution time, the models with only CCCT or CVCT features were faster to train than the one using both features.

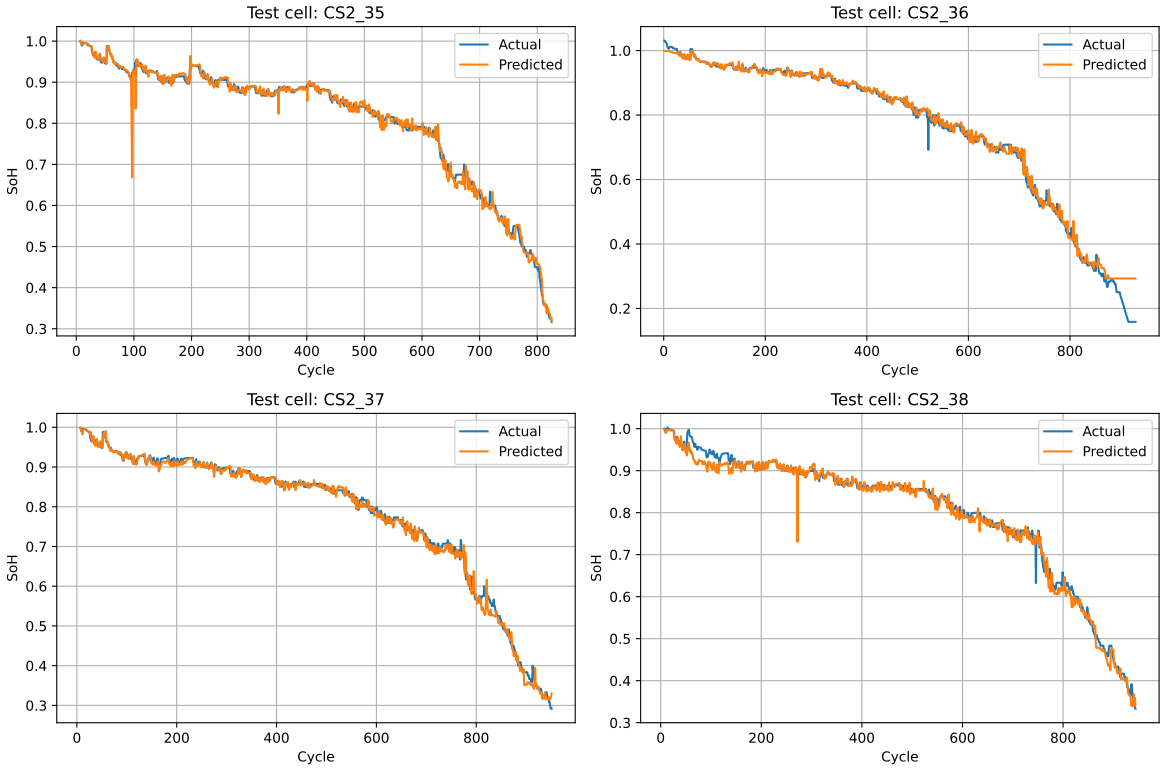


Figure 5.13: SoH estimation using Random Forest on dataset-2

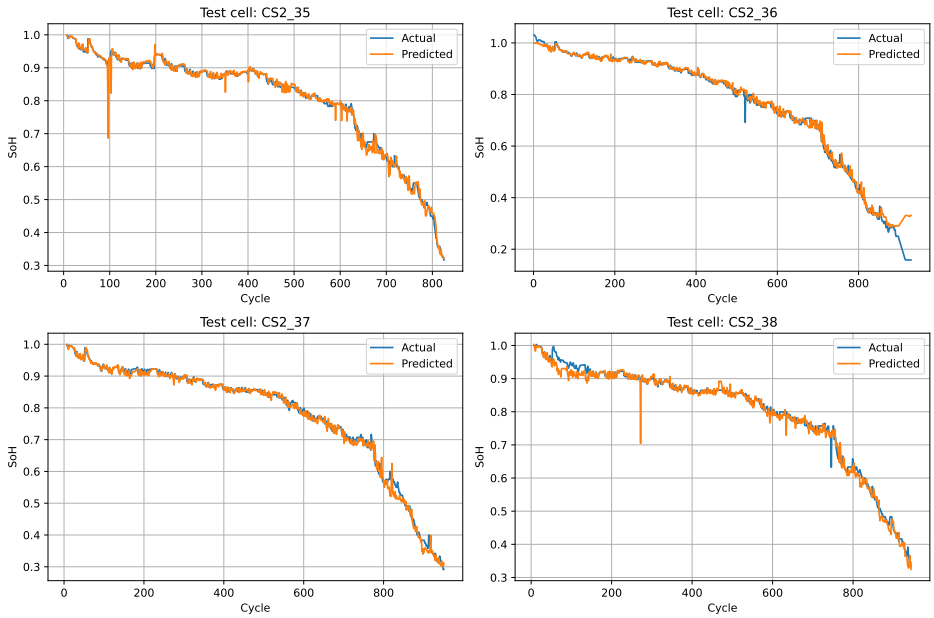


Figure 5.14: SoH estimation using XGBoost on dataset-2

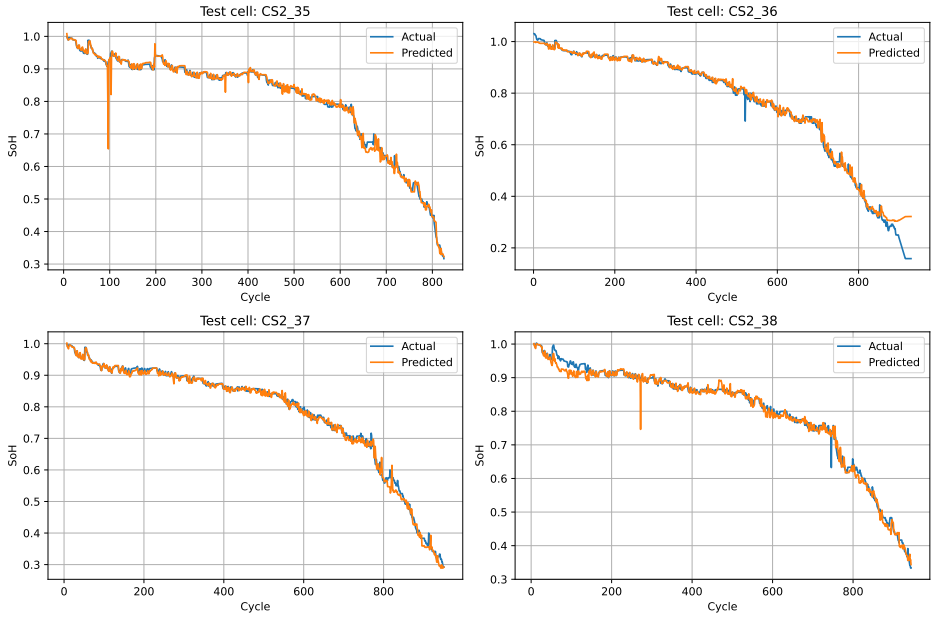


Figure 5.15: SoH estimation using LightGBM on dataset-2

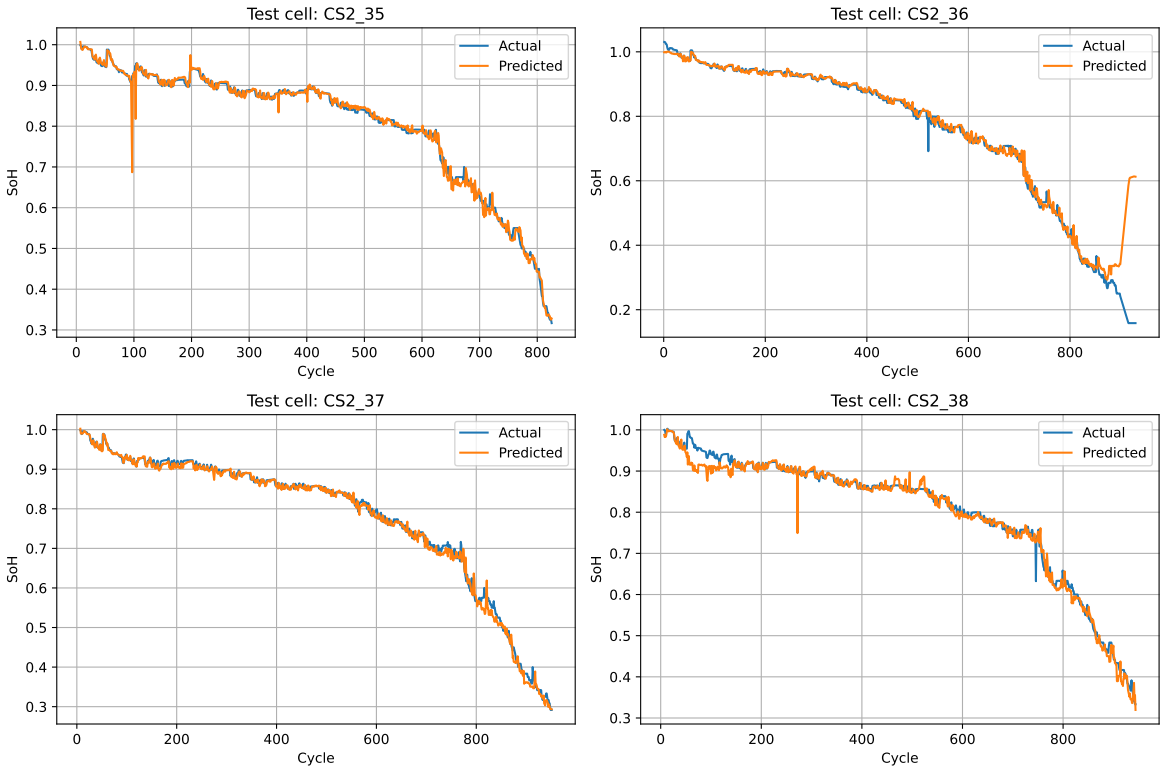


Figure 5.16: SoH estimation using Catboost on dataset-2

Figures 5.13, 5.14, 5.15, and 5.16 demonstrate the performance of the Random Forest, XGBoost, LightGBM, and CatBoost models, respectively. We can observe that the estimated values by the models are very close to the actual values. The predictions made by the models in this case are significantly better compared to the predictions in the case of dataset-1. This is mainly due to the increased no. of samples present in this datasets. So, the models had more data to train from.

In conclusion, all the models (Random Forest, XGBoost, LightGBM, and CatBoost) performed well in estimating the SoH values over cycles, as evidenced by the close alignment of the 'cycle vs predicted SoH' and 'cycle vs actual SoH' lines in the figures.

5.3.1 Hyperparameter Optimization

Table 5.13: Comparison of tuned Random Forest model performance for different feature selection methods on dataset-2

	mse	rmse	r2	mae	mape	time
base	0.000 237	0.015 225	0.991 565	0.009 792	0.015 482	4.568 123
CCCT	0.000 301	0.017 208	0.989 166	0.011 356	0.017 760	2.768 514
CVCT	0.009 274	0.093 456	0.684 215	0.059 173	0.105 549	2.753 624

Table 5.14: Comparison of tuned XGBoost model performance for different feature selection methods on dataset-2

	mse	rmse	r2	mae	mape	time
base	0.000 416	0.019 538	0.986 559	0.012 883	0.021 702	0.085 085
CCCT	0.000 410	0.019 477	0.986 619	0.012 885	0.021 568	0.064 161
CVCT	0.008 392	0.088 484	0.719 865	0.056 589	0.100 779	0.068 000

Table 5.15: Comparison of tuned LGBM model performance for different feature selection methods on dataset-2

	mse	rmse	r2	mae	mape	time
base	0.000 240	0.015 165	0.991 679	0.009 534	0.015 553	3.188 894
CCCT	0.000 307	0.017 243	0.989 235	0.011 456	0.018 343	1.184 303
CVCT	0.008 608	0.089 972	0.709 898	0.057 399	0.102 596	1.197 367

Table 5.16: Comparison of tuned Catboost model performance for different feature selection methods on dataset-2

	mse	rmse	r2	mae	mape	time
CCCT	0.000 334	0.017 679	0.989 026	0.011 396	0.018 937	1.138 117
CVCT	0.008 199	0.087 387	0.726 813	0.055 954	0.099 950	1.115 647

Tables 5.13, 5.14, 5.15, and 5.16 present a comparative evaluation of the performance metrics for the Random Forest, XGBoost, LightGBM, and CatBoost models after hyperparameter optimization, respectively.

In the case of the Random Forest model (Table 5.13), the performance across all metrics improved after optimization, particularly regarding a decrease in the Mean Squared Error (MSE) and an increase in the Coefficient of Determination (R^2). The Constant Current Charge Time (CCCT) feature yielded slightly better results than the Constant Voltage Charge Time (CVCT).

For the XGBoost model (Table 5.14), the optimized performance exhibited a minor increase in the metrics compared to the default model, with CCCT outperforming CVCT in terms of Root Mean Squared Error (RMSE) and R^2.

The LightGBM model (Table 5.15) also improved across all metrics, exhibiting the highest R^2 value with all features selected. However, there was no significant difference in the model's performance with CCCT and CVCT features after tuning.

Lastly, the CatBoost model (Table 5.16) showed improved performance after optimization, particularly with the CCCT feature, which outperformed both and CVCT feature in terms of MSE, RMSE, and R^2.

In conclusion, after hyperparameter optimization, all models showed improved performance. The models with both CCCT and CVCT selected as features generally performed well. Between the two features, CCCT appeared to yield slightly better results than CVCT.

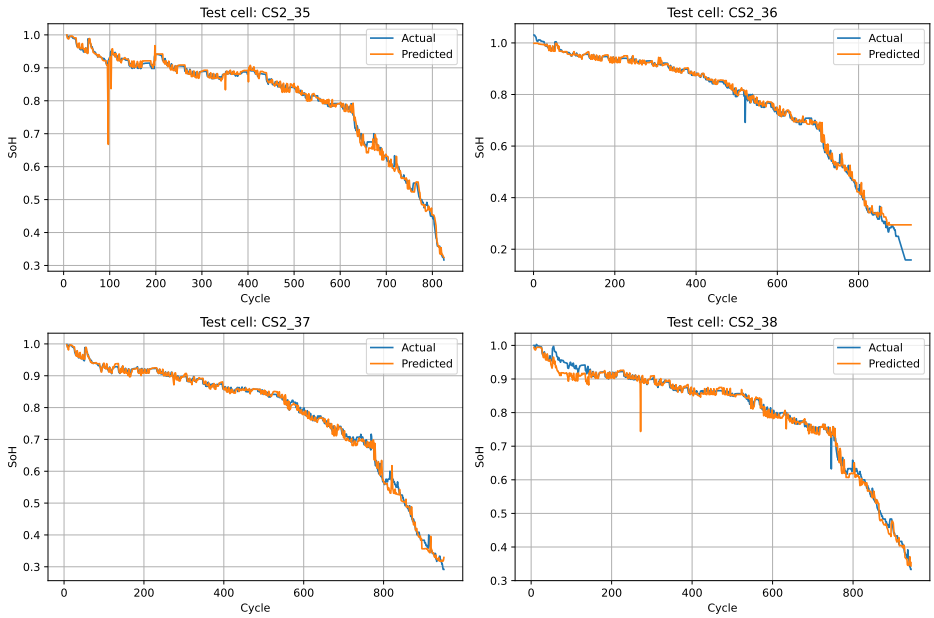


Figure 5.17: SoH estimation using Random Forest on dataset-2 after hyperparameter optimization

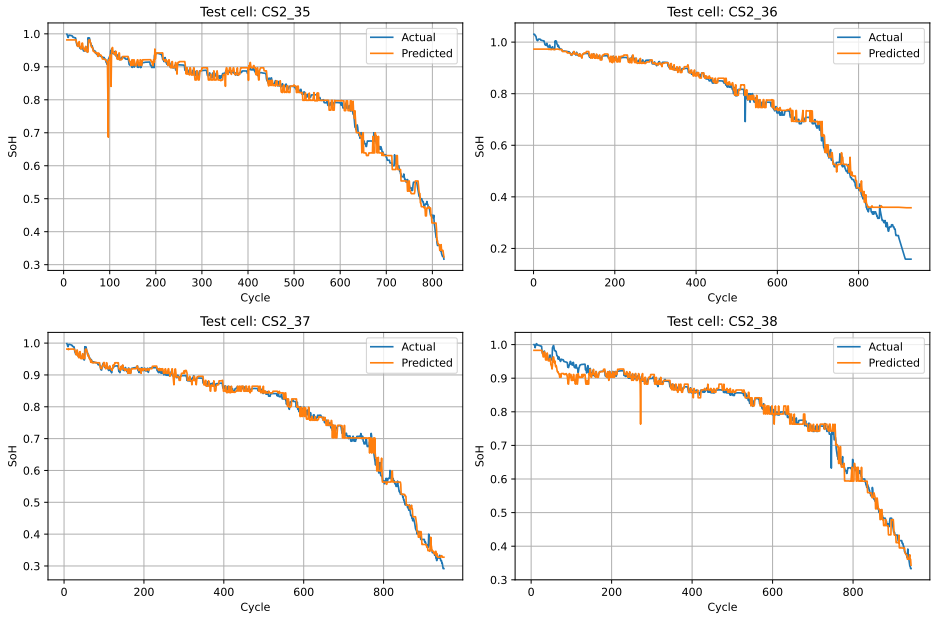


Figure 5.18: SoH estimation using XGBoost on dataset-2 after hyperparameter optimization

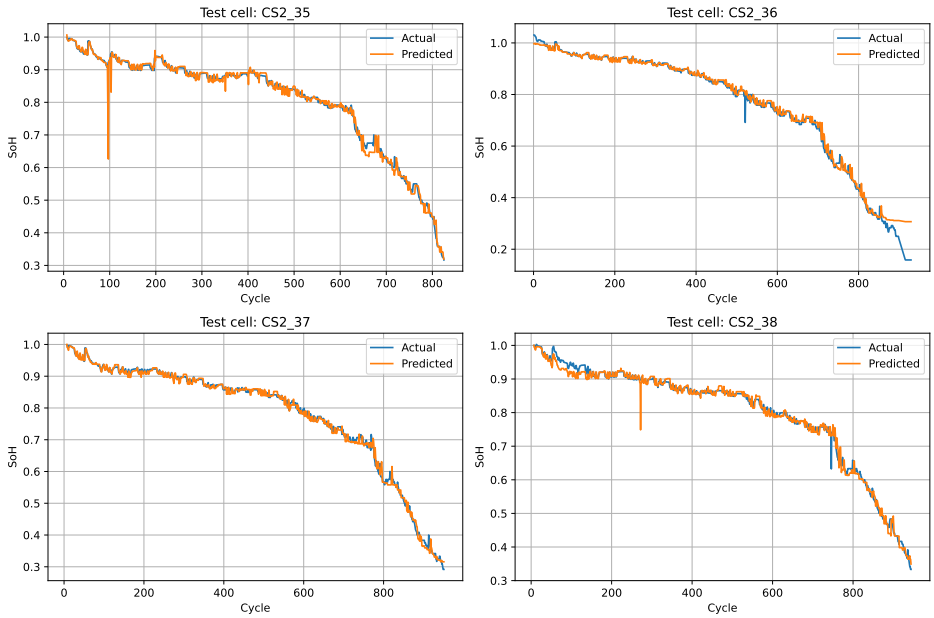


Figure 5.19: SoH estimation using LightGBM on dataset-2 after hyperparameter optimization

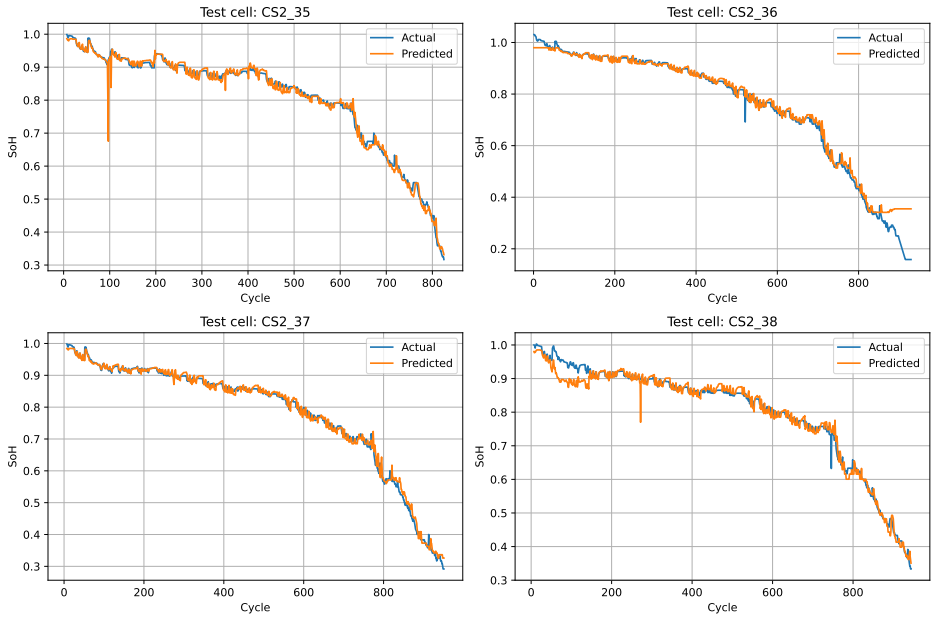


Figure 5.20: SoH estimation using Catboost on dataset-2 after hyperparameter optimization

Figures 5.17, 5.18, 5.19, and 5.20 demonstrate the improved performance of the Random Forest, XGBoost, LightGBM, and CatBoost models, respectively, on dataset-2 after hyper-

parameter optimization. In each figure, the 'cycle vs predicted SoH' line closely follows the 'cycle vs actual SoH' line, indicating enhanced predictive accuracy over cycles compared to the regular models.

Hyperparameter optimization process has led to noticeable improvements in the performance of all the models for this dataset. However, it takes quite a lot of time to tune the parameters which may increase the computational cost for model training. So, the choice of hyperparameter tuning depends on the size and the type of dataset.

CHAPTER 6

CONCLUSION

In conclusion, our research on enhancing Battery State of Health (SoH) estimation using ensemble and boosting techniques has demonstrated the effectiveness of these algorithms in developing accurate and reliable SoH estimation models for Li-ion batteries. Among the algorithms used, LightGBM had the best result in terms of accuracy and computation time. By employing Bayesian Optimization for hyperparameter tuning, we were able to improve the performance of the models. Additionally, the removal of insignificant data through feature selection helped focus on the most informative features, further enhancing the accuracy of the models. It also allowed us to save costs by reducing the amount of storage and sensors needed to capture the data.

The findings of our research have practical implications for various applications that rely on Li-ion batteries, such as electric vehicles, renewable energy systems, and consumer electronics. Accurately estimating the SoH of batteries is crucial for optimizing their performance, ensuring safety, and prolonging their lifespan. By employing these methods, we can enhance the accuracy of SoH estimation, enabling better management and utilization of Li-ion batteries in these applications.

Overall, our research contributes to the advancement of SoH estimation for Li-ion batteries and provides valuable insights for researchers and practitioners in the field. The application of recent and robust machine learning algorithms and the optimization technique can greatly benefit the performance and reliability of Li-ion batteries in real-world scenarios, paving the way for their wider adoption in various industries.

REFERENCES

[1] S. Li, H. Fang, and B. Shi, "Remaining useful life estimation of lithium-ion battery based on interacting multiple model particle filter and support vector regression," *Reliability Engineering & System Safety*, vol. 210, p. 107542, 2021.

[2] S. Ansari, A. Ayob, M. H. Lipu, A. Hussain, and M. H. M. Saad, "Remaining useful life prediction for lithium-ion battery storage system: A comprehensive review of methods, key factors, issues and future outlook," *Energy Reports*, vol. 8, pp. 12153–12185, 2022.

[3] N. Wassiliadis, J. Adermann, A. Frericks, M. Pak, C. Reiter, B. Lohmann, and M. Lienkamp, "Revisiting the dual extended kalman filter for battery state-of-charge and state-of-health estimation: A use-case life cycle analysis," *Journal of Energy Storage*, vol. 19, pp. 73–87, 2018.

[4] Y. Shi, S. Ahmad, Q. Tong, T. M. Lim, Z. Wei, D. Ji, C. M. Eze, and J. Zhao, "The optimization of state of charge and state of health estimation for lithium-ions battery using combined deep learning and kalman filter methods," *International Journal of Energy Research*, vol. 45, no. 7, pp. 11206–11230, 2021.

[5] F. Zhu and J. Fu, "A novel state-of-health estimation for lithium-ion battery via unscented kalman filter and improved unscented particle filter," *IEEE Sensors Journal*, vol. 21, no. 22, pp. 25449–25456, 2021.

[6] M. Zeng, P. Zhang, Y. Yang, C. Xie, and Y. Shi, "Soc and soh joint estimation of the power batteries based on fuzzy unscented kalman filtering algorithm," *Energies*, vol. 12, no. 16, p. 3122, 2019.

[7] D. Liu, X. Yin, Y. Song, W. Liu, and Y. Peng, "An on-line state of health estimation of lithium-ion battery using unscented particle filter," *Ieee Access*, vol. 6, pp. 40990–41001, 2018.

[8] L. Ungurean, G. Cârstoiu, M. V. Micea, and V. Groza, "Battery state of health estimation: a structured review of models, methods and commercial devices," *International Journal of Energy Research*, vol. 41, no. 2, pp. 151–181, 2017.

[9] Z. Guo, X. Qiu, G. Hou, B. Y. Liaw, and C. Zhang, "State of health estimation for lithium ion batteries based on charging curves," *Journal of Power Sources*, vol. 249, pp. 457–462, 2014.

[10] C. Weng, J. Sun, and H. Peng, "A unified open-circuit-voltage model of lithium-ion batteries for state-of-charge estimation and state-of-health monitoring," *Journal of power Sources*, vol. 258, pp. 228–237, 2014.

[11] S. S. Sheikh, M. Anjum, M. A. Khan, S. A. Hassan, H. A. Khalid, A. Gastli, and L. Ben-Brahim, "A battery health monitoring method using machine learning: A data-driven approach," *Energies*, vol. 13, no. 14, p. 3658, 2020.

[12] Z. Liu, J. Zhao, H. Wang, and C. Yang, "A new lithium-ion battery soh estimation method based on an indirect enhanced health indicator and support vector regression in phms," *Energies*, vol. 13, no. 4, p. 830, 2020.

[13] X. Bian, Z. Wei, J. He, F. Yan, and L. Liu, "A novel model-based voltage construction method for robust state-of-health estimation of lithium-ion batteries," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 12, pp. 12173–12184, 2020.

[14] X. Shu, G. Li, J. Shen, Z. Lei, Z. Chen, and Y. Liu, "A uniform estimation framework for state of health of lithium-ion batteries considering feature extraction and parameters optimization," *Energy*, vol. 204, p. 117957, 2020.

[15] X. Shu, G. Li, J. Shen, W. Yan, Z. Chen, and Y. Liu, "An adaptive fusion estimation algorithm for state of charge of lithium-ion batteries considering wide operating temperature and degradation," *Journal of Power Sources*, vol. 462, p. 228132, 2020.

[16] X. Hu, Y. Che, X. Lin, and S. Onori, "Battery health prediction using fusion-based feature selection and machine learning," *IEEE Transactions on Transportation Electrification*, vol. 7, no. 2, pp. 382–398, 2020.

[17] Y. Deng, H. Ying, E. Jiaqiang, H. Zhu, K. Wei, J. Chen, F. Zhang, and G. Liao, "Feature parameter extraction and intelligent estimation of the state-of-health of lithium-ion batteries," *Energy*, vol. 176, pp. 91–102, 2019.

[18] Z. Deng, X. Hu, X. Lin, L. Xu, Y. Che, and L. Hu, "General discharge voltage information enabled health evaluation for lithium-ion batteries," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 3, pp. 1295–1306, 2020.

[19] H. Xu, Y. Peng, and L. Su, "Health state estimation method of lithium ion battery based on nasa experimental data set," in *IOP Conference Series: Materials Science and Engineering*, vol. 452, p. 032067, IOP Publishing, 2018.

[20] L. Song, K. Zhang, T. Liang, X. Han, and Y. Zhang, "Intelligent state of health estimation for lithium-ion battery pack based on big data analysis," *Journal of Energy Storage*, vol. 32, p. 101836, 2020.

[21] Y. Li, S. Zhong, Q. Zhong, and K. Shi, "Lithium-ion battery state of health monitoring based on ensemble learning," *IEEE access*, vol. 7, pp. 8754–8762, 2019.

[22] S. Song, C. Fei, and H. Xia, "Lithium-ion battery soh estimation based on xgboost algorithm with accuracy correction," *Energies*, vol. 13, no. 4, p. 812, 2020.

[23] Y. Choi, S. Ryu, K. Park, and H. Kim, "Machine learning-based lithium-ion battery capacity estimation exploiting multi-channel charging profiles," *Ieee Access*, vol. 7, pp. 75143–75152, 2019.

[24] Y. Li, C. Zou, M. Berecibar, E. Nanini-Maury, J. C.-W. Chan, P. Van den Bossche, J. Van Mierlo, and N. Omar, "Random forest regression for online capacity estimation of lithium-ion batteries," *Applied energy*, vol. 232, pp. 197–210, 2018.

[25] Y. Li, D.-I. Stroe, Y. Cheng, H. Sheng, X. Sui, and R. Teodorescu, "On the feature selection for battery state of health estimation based on charging–discharging profiles," *Journal of Energy Storage*, vol. 33, p. 102122, 2021.

[26] X. Shu, G. Li, Y. Zhang, J. Shen, Z. Chen, and Y. Liu, "Online diagnosis of state of health for lithium-ion batteries based on short-term charging profiles," *Journal of Power Sources*, vol. 471, p. 228478, 2020.

[27] H. Sheng, X. Liu, L. Bai, H. Dong, and Y. Cheng, "Small sample state of health estimation based on weighted gaussian process regression," *Journal of Energy Storage*, vol. 41, p. 102816, 2021.

[28] J. Jiang, S. Zhao, and C. Zhang, "State-of-health estimate for the lithium-ion battery using chi-square and elm-lstm," *World Electric Vehicle Journal*, vol. 12, no. 4, p. 228, 2021.

[29] X. Li, C. Yuan, and Z. Wang, "State of health estimation for li-ion battery via partial incremental capacity analysis based on support vector regression," *Energy*, vol. 203, p. 117852, 2020.

[30] B. Xiao, B. Xiao, and L. Liu, "State of health estimation for lithium-ion batteries based on the constant current–constant voltage charging curve," *Electronics*, vol. 9, no. 8, p. 1279, 2020.

[31] Q. Li, D. Li, K. Zhao, L. Wang, and K. Wang, "State of health estimation of lithium-ion battery based on improved ant lion optimization and support vector regression," *Journal of Energy Storage*, vol. 50, p. 104215, 2022.

[32] Z. Chen, Q. Xue, R. Xiao, Y. Liu, and J. Shen, "State of health estimation for lithium-ion batteries based on fusion of autoregressive moving average model and elman neural network," *IEEE access*, vol. 7, pp. 102662–102678, 2019.

[33] Y. Wu, Q. Xue, J. Shen, Z. Lei, Z. Chen, and Y. Liu, "State of health estimation for lithium-ion batteries based on healthy features and long short-term memory," *IEEE Access*, vol. 8, pp. 28533–28547, 2020.

[34] Y. Li, H. Sheng, Y. Cheng, D.-I. Stroe, and R. Teodorescu, "State-of-health estimation of lithium-ion batteries based on semi-supervised transfer component analysis," *Applied Energy*, vol. 277, p. 115504, 2020.

[35] J. He, Z. Wei, X. Bian, and F. Yan, "State-of-health estimation of lithium-ion batteries using incremental capacity analysis based on voltage–capacity model," *IEEE Transactions on Transportation Electrification*, vol. 6, no. 2, pp. 417–426, 2020.

[36] S. Zhang, B. Zhai, X. Guo, K. Wang, N. Peng, and X. Zhang, "Synchronous estimation of state of health and remaining useful lifetime for lithium-ion battery using the incremental capacity and artificial neural networks," *Journal of Energy Storage*, vol. 26, p. 100951, 2019.

[37] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[38] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

[39] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, 2017.

[40] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," *Advances in neural information processing systems*, vol. 31, 2018.

[41] "Prognostics center of excellence data set repository." urlhttps://www.nasa.gov/content/prognostics-center-of-excellence-data-set-repository. Accessed: 2023-04-28.

[42] "Battery data." urlhttps://calce.umd.edu/battery-data. Accessed: 2023-04-28.

[43] G. Dos Reis, C. Strange, M. Yadav, and S. Li, "Lithium-ion battery data and where to find it," *Energy and AI*, vol. 5, p. 100081, 2021.

[44] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*, pp. 413–422, IEEE, 2008.

[45] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[46] S. Raschka, "Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack," *Journal of open source software*, vol. 3, no. 24, p. 638, 2018.

[47] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[48] "Scikit-optimize." urlhttps://scikit-optimize.github.io/stable/. Accessed: 2023-04-28.