People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

Djilali Bounaama Khemis Miliana University

Faculty of Science and Technology

Department of Technology

Presented Thesis

For the Purpose of Obtaining a Degree

Master's Degree

in AUTOMATION

Specialization: Automation and Industrial Computing

Theme

Smart method for Battery State Prediction (State of Charge) through battery data analysis

Presented by

Mr. GUIDJELLI Abdelhafid Amar

Jury:

Mr. FEKIR	UDBKM	President
Mr. HOUCIN	UDBKM.	Examiner
Mr. MERROUCHE Walid	CDER	Supervisor
Mr. BLAIFI Sidali	UDBKM	Co-Supervisor

Academic Year 2022/2023

DEDICATION

I have succeeded in my studies.

This achievement is a sincere tribute to my beloved Grandfather Abdel Latife. May the Almighty God keep you and bless you with good health, happiness, and a long life.

To my incredible parents

My father, who always believed in me and provided me with all the necessary means for my education,

and my mother, who is kind, honorable, and amiable. You both symbolize goodness, are a source of tenderness, and have never ceased to encourage me and pray for me.

To my cherished brothers

Yacine, Mohcine, and Mohamed and their wives, not forgetting the children Fatima and Abdellatife.

I extend a special dedication to my sister Rihane, who has supported me throughout this journey. I am grateful for your unwavering support. May Allah grant you a long life and fulfill all your wishes.

To my aunts and uncles

thank you for the love you have given me and for your precious encouragement.

To my cousins and all my family members Your support has meant the world to me.

To my supportive friends Amine, Walid, Alae,

and all the friends from Automation and Industrial Computing Thank you for the wonderful times we have spent together. Your friendship and companionship have been invaluable.

Lastly, I want to express my gratitude to all those who have helped and guided me along the way. Each of you has made an indelible mark on my life, offering love, support, and inspiration. Your presence has been a source of strength and joy, and I am forever grateful for the invaluable role you have played in shaping the person I am today.

GUIDJELLI ABDELHAFID AMAR

THANKS

First and foremost, I express my gratitude to God for granting us the courage and good health to complete this project.

This endeavor would not have been possible without the assistance and encouragement of several individuals, to whom i extend my heartfelt thanks.

I would like to express my deep appreciation to Mr. Merrouch for his unwavering support, relevant advice, and for providing me with the subject of this work. It is through his efforts that i was able to undertake this research.

I would also like to acknowledge Mr. Belaifi for his invaluable guidance, supervision, and meticulous review of my work. His assistance has been instrumental in shaping the outcome of this thesis.

Furthermore, I extend my gratitude to the members of the jury, who have graciously participated and diligently evaluated the content of this thesis. I am honored by their presence and thankful for their thorough examination.

I would like to express my utmost respect and thanks to the entire faculty of the Department of Science and Technology at the University of Djilali Bounaama in Khemis Miliana. Their dedication and expertise have greatly contributed to my education and training in this field.

Lastly, I want to sincerely thank all those who have directly or indirectly contributed to the development of this work. Your involvement has played an integral role in its success.

Abstract

Estimating battery charge status remains a challenge in verifying the actual charge level. Researchers are still working on developing new systems to estimate charge status and battery health. Regarding the problem of the inability to directly measure the state of charge of a lithium-ion battery, this thesis provides a comprehensive overview of charge status and battery management systems, including various existing methods to estimate this charge status. It specifically proposes a method for estimating charge status using artificial intelligence, specifically random forest regression. Firstly, a training set was created using battery current, voltage, temperature, average voltage, and average current as inputs to the model, with the corresponding battery charge state as the output. Then, the model was trained using the random forest regression algorithm. The study successfully achieved accurate predictions of charge status using machine learning techniques, contributing to the development of battery management systems and highlighting the potential of random forest technology for accurate charge status estimation.

keywords: state of charge (SOC), AI ,ML, Random Forest Regression, decision trees, BMS.

ملخص

لا يزال تقدير حالة شحن البطارية مشكل في التحقق من مقدار شحن البطارية الحقيقي، و ما زال الباحثون الى حد الان يعملون على تطوير انظمة جديدة لتقدير حالة شحن وصحة البطارية. بالنسبة لمشكلة عدم إمكانية قياس حالة شحن بطارية الليثيوم أيون بشكل مباشر، سنقدم نظرة شاملة في هذه المذكرة عن حالة شحن البطارية ونظام ادارتها، مع ذكر مختلف الطرق الموجودة لتقدير حالة شحن البطارية. وسنتخصص باقتراح طريقة تقدير حالة شحن البطارية باستخدام الذكاء الاصطناعي، وتحديدا الانحدار العشوائي للغابات. أولاً، تم إنشاء مجموعة تدريب تستخدم التيار، الجهد الكهربائي، درجة الحرارة، متوسط الجهد الكهربائي ومتوسط التيار للبطارية، كمدخل تدريب للنموذج. وحالة شحن للبطارية المقابل كمخرج تدريب للنموذج. بعد ذلك، تم تدريب النموذج باستخدام خوارزمية الغابة العشوائية للانحدار. تمكنت الدراسة من تحقيق توقع جد دقيق لحالة الشحن من خلال استخدام تقنيات التعلم الآلي، مما يساهم في تطوير نظم إدارة البطاريات ويسلط الضوء على إمكانية التقنية العشوائية للتنبؤ في مجال تقدير حالة الشحن بشكل دقيق.

الكلمات المفتاحية: حالة الشحن، الذكاء الاصطناعي، التعلم الآلي، الغابات العشوائية، شجرة القرار، نظام إدارة البطاريات.

Résumé

L'estimation de l'état de charge de la batterie reste un défi pour vérifier le niveau de charge réel. Les chercheurs travaillent encore sur le développement de nouveaux systèmes pour estimer l'état de charge et la santé de la batterie. En ce qui concerne le problème de l'impossibilité de mesurer directement l'état de charge d'une batterie lithium-ion, ce mémoire présente un aperçu complet de l'état de charge et des systèmes de gestion de la batterie, y compris les différentes méthodes existantes pour estimer cet état de charge. Elle propose spécifiquement une méthode d'estimation de l'état de charge en utilisant l'intelligence artificielle, plus précisément la régression de la forêt aléatoire. Tout d'abord, un ensemble d'entraînement a été créé en utilisant le courant, la tension, la température, la tension moyenne et le courant moyen de la batterie comme entrée du modèle, avec l'état de charge de la batterie correspondant comme sortie. Ensuite, le modèle a été entraîné en utilisant l'algorithme de régression de la forêt aléatoire. L'étude a réussi à obtenir des prédictions précises de l'état de charge en utilisant des techniques d'apprentissage automatique, ce qui contribue au développement de systèmes de gestion de batterie et met en évidence le potentiel de la technologie de la forêt aléatoire pour une estimation précise de l'état de charge.

Mots-clés : état de charge (SOC), IA, apprentissage automatique, régression par forêts aléatoires, arbres de décision, système de gestion de batteries.

Contents

List of Figures

List of Abbreviations

List of Abbreviations

SOC	State of Charge
IA	Intelligence Artificielle (Artificial Intelligence)
ML	Machine Learning
BMS	Battery Management System
ECM	Equivalent Circuit Model
MSE	Mean Squared Error
SOH	State of Health
EVs	Electric Vehicles
OCV	Open Circuit Voltage
UPS	Uninterruptible Power Supply
SOP	State of Power
HEVs	Hybrid Electric Vehicles
PCA	Principal Component Analysis
t-SNE	t-Distributed Stochastic Neighbor Embedding
FNN	Feedforward Neural Network
HTML	HyperText Markup Language
Li-ion	Lithium-ion
GPU	Graphics Processing Unit
TPU	Tensor Processing Unit

General Introduction

Battery state prediction, specifically the estimation of state of charge (SOC), is a critical aspect of battery management systems in various applications, such as electric vehicles and renewable energy systems. Accurate SOC estimation enables efficient battery utilization, optimal system performance, and reliable operation. However, accurately predicting battery SOC is a complex task due to the dynamic nature of battery behavior and the influence of various factors.

In this context, our research focuses on developing a smart method for battery state prediction through battery data analysis. We aim to leverage machine learning techniques, particularly Random Forest Regression, to overcome the challenges associated with SOC estimation and improve the accuracy and reliability of battery state prediction.

To address this research problem, our work is organized into three chapters.

The first chapter provides a comprehensive introduction to the concept of SOC. We discuss its definition, various uses in battery management systems, and the benefits and importance of accurate SOC estimation. Additionally, we explore different SOC estimation methods, including coulomb counting, voltage-based methods, Kalman filter-based methods, and the equivalent circuit model (ECM). Understanding the fundamentals of SOC estimation sets the foundation for our subsequent chapters.

In the second chapter, we delve into the key concepts of artificial intelligence and machine learning. We explore different types of machine learning, such as supervised and unsupervised learning, and highlight the relevance of decision trees as a fundamental concept. Moreover, we introduce random forests, an ensemble learning technique known for its ability to handle complex problems and deliver accurate predictions. Understanding these machine learning concepts is essential for developing our battery state prediction model.

The third chapter focuses on the practical aspects of our research. We discuss the preprocessing steps performed on the battery data, including the conversion of file formats, handling missing values, outlier detection and treatment, and feature scaling and normalization. Subsequently, we delve into the process of model training using Random Forest Regression. We present the methodology and tools used, including Python and Google Colab, and describe the steps involved in preparing the training data, splitting the data into training, validation, and test sets, and training the model. Finally, we present the results of our model training, including evaluation metrics such as mean squared error (MSE) and R-squared, and analyze and interpret the findings.

Finally , our research focuses on developing a smart method for battery state prediction using machine learning techniques, particularly Random Forest Regression. By addressing the research questions and exploring the key concepts and methodologies involved in SOC estimation, we aim to enhance the accuracy and reliability of battery state prediction. Through the organization of our work into three chapters, we provide a comprehensive framework for understanding the general state of charge, machine learning concepts, and the practical implementation of our battery state prediction method.

Chapter 1

Overview State of Charge Estimation

1.1 Introduction

In this chapter, we will delve into the significance and various aspects of SOC and BMS in the realm of batteries and energy storage systems. SOC is a crucial parameter that measures the remaining energy capacity of a battery, and BMS plays a pivotal role in monitoring and controlling battery performance. We will explore the definitions of SOC and BMS, highlighting their uses and discussing the benefits and importance they offer. Additionally, we will examine different types of SOC estimation methods, including Coulomb Counting, Voltage-based methods, Kalman filter-based methods, and the Equivalent Circuit Model (ECM). Each method has its own advantages and applications.

1.2 State of Charge (SOC)

1.2.1 Definition

The State of Charge (SOC) is a measure of the amount of charge present in a battery at a given time, expressed as a percentage of the battery's maximum capacity. It indicates the amount of energy stored in the battery and is an important parameter to determine the remaining runtime of a battery-powered device. [1, 10]

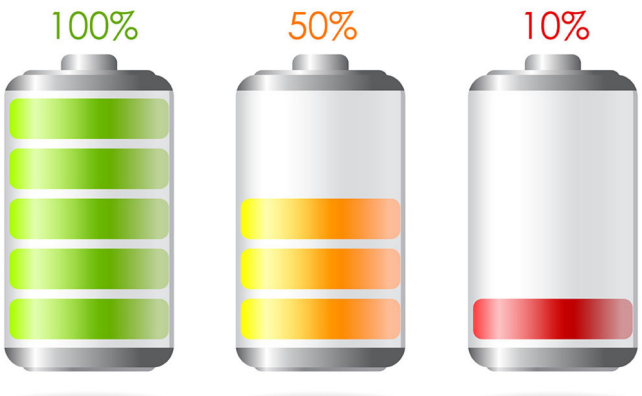

Figure 1.1: State of charge [23]

On the other words, SOC represents the current energy level of a battery, which is usually expressed as a percentage of its total energy capacity. For example, if a battery has a total energy capacity of $100Wh$ (watt-hours) and its current SOC is 50, it means that it currently has $50Wh$ of energy stored in it. SOC is a crucial parameter in battery management systems as it enables the estimation of the remaining runtime of a battery-powered device and ensures that the battery is used efficiently without be damaged.The SOC can be defined as follows 1.1 [1]:

$$\mathrm{SOC}(t) = \frac{Q(t)}{Q_n} \tag{1.1}$$

Moreover, the battery efficinecy is significantly influenced by factors such as aging, working temperature, and the number cycles, thereby posing a formidable challenge for the sake to obtain an accurate estimat of the SOC [2].

1.2.2 State of Charge applications

The SOC has various uses in different applications, including:

• **Battery Management Systems (BMS):** In electric vehicles and hybrid electric vehicles, the BMS uses SOC to determine the remaining range of the vehicle and to prevent overcharging or over-discharging of the battery, which can cause damage and reduce its lifespan [3].

• **Portable Electronic Devices:** In portable electronic devices such as smartphones, tablets, and laptops, SOC is used to estimate the remaining runtime of the device and to provide users with an indication of when they need to recharge their battery. [4]

• **Renewable Energy Storage:** In renewable energy systems such as solar panels and wind turbines, SOC is used to manage the charging and discharging of batteries, ensuring that the energy is stored and used efficiently. [5]

• **Aerospace Applications:** In aerospace applications such as satellites and spacecraft, SOC is used to monitor and manage the battery's performance and health, ensuring that the battery can deliver power reliably over long periods. [6]

• **Medical Devices:**In medical devices such as implantable cardiac pacemakers and defibrillators, SOC is used to monitor the battery's performance and ensure that it has enough energy to operate safely and effectively. [7] So, SOC is a critical parameter in battery-powered ensures efficient battery use,prevents damage to the battery.

1.2.3 Benefits and Importance

The SOC is a critical parameter in battery management systems inorder to determine the battery state at any given time. Its usefulness and importance can be summarized as follows:

• **Estimation of Remaining Runtime:** SOC is used to estimate the remaining runtime of a battery-powered device accurately. This information is essential which allows the user to plan their activities and ensures that they do not run out of battery power unexpectedly. [10]

• **Prevention of Overcharging and Over-Discharging:** SOC is used to prevent overcharging and over-discharging of batteries, which can cause damage and reduce their lifespan. Battery management systems use SOC to regulate the charging and discharging of batteries to ensure that they operate within the safe limits. [9]

• **Optimization of Battery Use:** SOC is used to optimize battery use by providing information about the energy available in a battery. This enables users to use their devices efficiently and reduces energy wastage. [2, 10]

• **Maintenance of Battery Health:** SOC is used to monitor the health of batteries, which enable to detect any degradation or issues with the battery's performance. This information can help take a corrective measures for the sake to maintain the battery's health and extend its lifespan. [10]

• **Safety:** SOC is used to ensure the safety of battery-powered devices and applications. Overcharging or over-discharging can damage the battery and leading to the risk of fire or explosion. By monitoring SOC, battery management systems can prevent such incidents and ensure the safe operation of battery-powered devices. [10]

In conclusion, SOC is a crucial parameter in battery management systems and battery-powered devices as it provides critical information about the battery's energy level, enabling users to estimate remaining runtime, optimize battery use, prevent damage, maintain battery health, and ensure safety.

1.3 Battery Management Systems (BMS)

1.3.1 Definition

A Battery Management System (BMS) is an electronic system that manages and monitors the performance of a battery pack. In the case of a battery solar lithium-ion system, the BMS is responsible for managing and monitoring the performance of the lithium-ion batteries used in the system. [9]

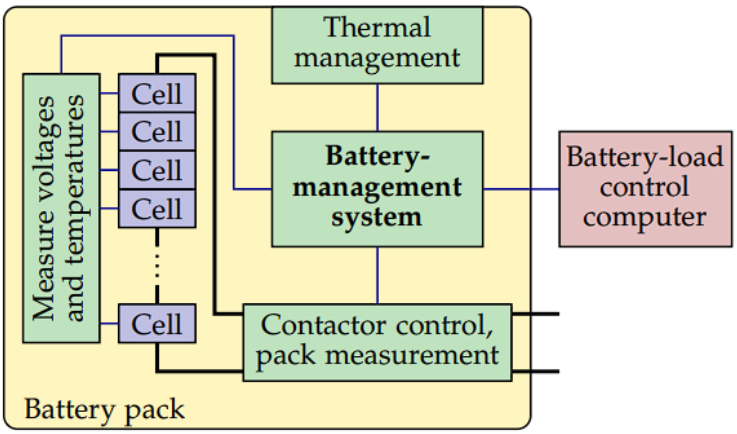

Figure 1.2: Context of BMS inside battery pack [9]

The BMS performs various functions, including monitoring the State of Charge (SOC) and State of Health (SOH) of the batteries, balancing the cells, controlling the charging and discharging of the batteries, temperature management, and overcurrent protection. [10]

The BMS is essential to ensure the safe and efficient operation of the batteries, extending their lifespan, and optimizing their performance. By constantly monitoring the performance of the batteries, the BMS avoid prevent damage, ensures that the batteries operate within safe limits, and alerts the user if any issues arise. The BMS also helps optimize the performance of

the batteries by controlling the charging and discharging process and balancing the cells within the battery pack, which can improve the battery's efficiency and extend its lifespan. [8]

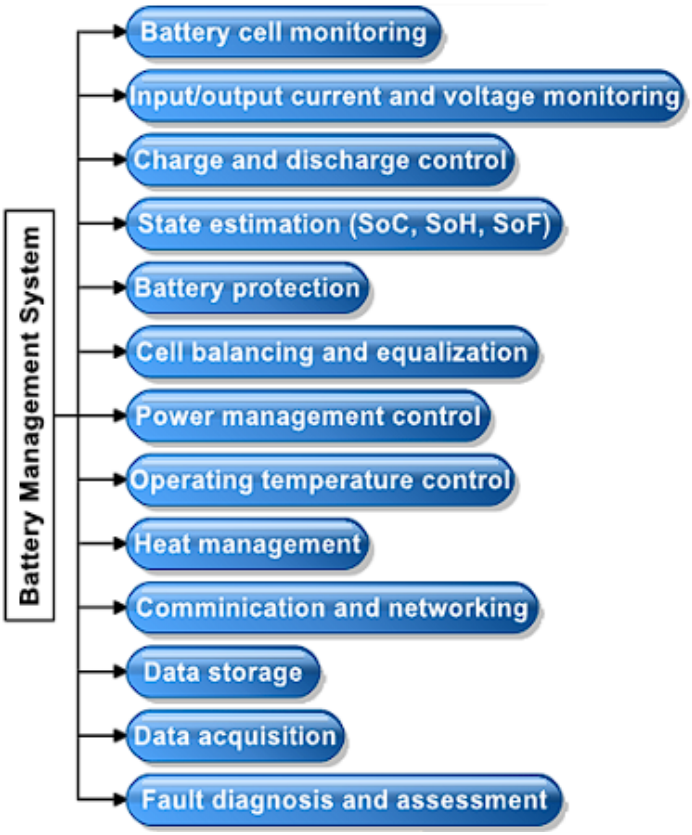

Figure 1.3: Aspects of the Battery Management System

1.3.2 Various uses of BMS

The Battery Management System (BMS) of a battery solar lithium-ion system has various uses, including:

• **Monitoring the State of Charge (SOC) and State of Health (SOH) of the batteries:**Monitoring the State of Charge (SOC) and State of Health (SOH) of the batteries: The BMS constantly monitors the SOC and SOH of the batteries, ensuring that they are operating within safe limits and alerting the user if any issues arise. [8, 10]

• **Balancing the cells:** The BMS ensures that the cells within the battery pack are balanced, preventing overcharging or over-discharging of individual cells, which can cause damage and reduce the lifespan of the battery. [8, 9]

• **Controlling the charging and discharging of the batteries:** The BMS controls the charging and discharging of the batteries to ensure that they operate within safe limits and optimize their performance. [8, 9]

• **Protection:** The protection system for battery-powered systems and battery packs necessitates the integration of electronic components and intelligent logic. This system serves to safeguard both the operator of the system and the battery pack itself against various potential hazards, including overcharge, overdischarge, overcurrent, cell short circuits, and extreme temperatures. [9]

• **Temperature management:** The BMS monitors the temperature of the battery pack,

ensuring that it does not overheat, which can cause damage to the batteries. [8, 9]

• **Overcurrent protection:** The BMS provides overcurrent protection, ensuring that the batteries are not subjected to excessive current, which can damage them. [8, 9]

• **Predictive maintenance:** The BMS can analyze the performance data of the batteries and predict when maintenance is required, allowing for proactive maintenance rather than reactive maintenance. [8, 9]

• **Data logging:** The BMS can record and store data on the performance of the batteries, allowing for analysis of their performance over time and identification of any trends or issues. [9]

1.3.3 Benefits and Importance

The Battery Management System (BMS) is essential in ensuring the safe and efficient operation of a battery solar lithium-ion system. It helps prevent damage to the batteries, ensures that they operate within safe limits, and alerts the user if any issues arise. The BMS also helps optimize the performance of the batteries by controlling the charging and discharging process and balancing the cells within the battery pack, which can improve the battery's efficiency and extend its lifespan. [10]

1.3.4 Advantages and drowbuck

1.3.4.1 Advantages

The BMS has both positives and negatives. Some positives of the BMS include:

• **Safety:** The BMS helps ensure the safe operation of the batteries, preventing overcharging, over-discharging, overheating, and other issues that can cause damage or be dangerous. [9]

• **Efficiency:** The BMS optimizes the performance of the batteries, improving their efficiency and extending their lifespan. [8]

• **Reliability:** The BMS constantly monitors the performance of the batteries, ensuring that they operate within safe limits and alerting the user if any issues arise, making the system more reliable. [10]

1.3.4.2 Drowbuck

However, there are also some negatives of the BMS, including:

• **Cost:** The BMS can be expensive, adding to the overall cost of the battery lithium-ion system. [11]

• **Complexity:** The BMS is a complex system that requires specialized knowledge and training to operate and maintain.

• **Performance limitations:** The BMS can limit the performance of the batteries, as it controls the charging and discharging process and balances the cells within the battery pack.

Overall, the positives of the BMS outweigh the negatives, as it is essential in ensuring the safe and efficient operation of a battery lithium-ion system.

1.4 State of Charge Estimation methodes

There are several types of State of Charge measurement techniques, each with its own advantages and disadvantages. The most common types of SOC measurement techniques are:

1.4.1 Coulomb Counting SOC

1.4.1.1 Definition

The Coulomb counting method is a commonly used technique for estimating the state of charge (SoC) of a battery, including lithium-ion batteries used in solar systems. The method is based on the principle of charge conservation, which states that the total charge entering or leaving the battery should be equal to the change in the battery's stored charge. In other words, the SoC can be calculated by integrating the current flowing into or out of the battery over time2.1 [1]:

$$SOC(t) = SOC(t-1) + \frac{I(t)}{Q_n} \cdot \Delta t \tag{1.2}$$

where:
$SOC(t)$:represents the state of charge at time t.
$SOC(t-1)$:represents the state of charge at the previous time, which is $t-1$.
$I(t)$:represents the current at time t.
Q_n : represents the nominal charge.
Δt : represents the elapsed time duration.

The equation states that the state of charge at time t is equal to the state of charge at the previous time$t1$, plus the amount added based on the current $I(t)$ divided by the nominal charge Q_n and multiplied by the time duration Δt [1]

The Coulomb counting method requires an accurate measurement of the battery's current and voltage, which can be obtained using current and voltage sensors. The method assumes that the battery's capacity remains constant over time, which is not always true in practice. Battery capacity can degrade over time due to factors such as temperature, aging, and cycling, (cycle life) which can lead to errors in the Coulomb counting method. As a result, the method can provide a reasonably accurate SoC estimate in the short term, but over time, errors can accumulate, leading to inaccuracies. [13]

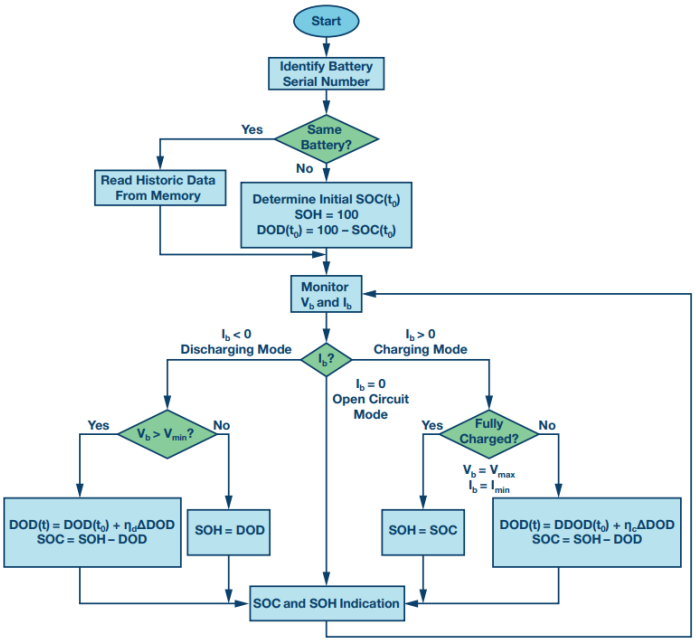

Figure 1.4: Flowchart of the enhanced coulomb counting algorithm. [12]

To improve the accuracy of the Coulomb counting method, various techniques can be used, including:

• **Temperature compensation:** Temperature can affect the battery's capacity, and therefore, the accuracy of the Coulomb counting method. Temperature sensors can be used to compensate for temperature changes and adjust the SoC estimate accordingly. [1]

• **Capacity estimation:** The battery's capacity can be estimated by periodically performing a discharge test and comparing the measured capacity with the rated capacity. The estimated capacity can be used to correct the Coulomb counting method's errors. [1]

• **Kalman filtering:** The Coulomb counting method can be combined with a Kalman filter algorithm to improve the accuracy of the SoC estimate. The Kalman filter can use information from other sensors, such as voltage, temperature, and current, to adjust the SoC estimate and reduce errors. [1]

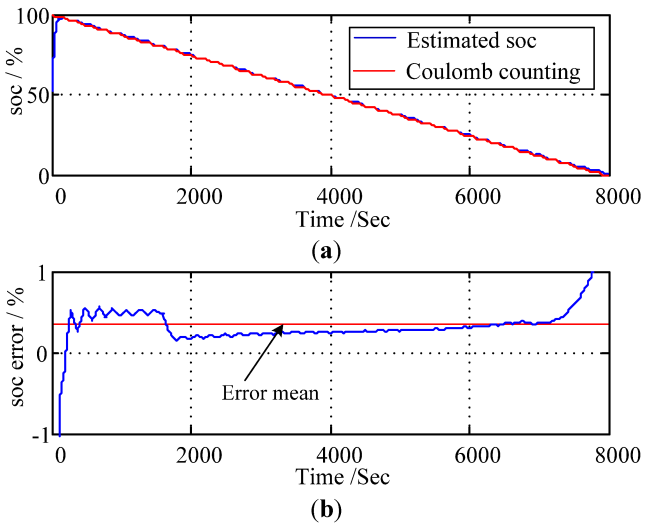

Figure 1.5: Estimation results and their error: (a) SOC estimation results; (b) SOC estimation errors. [17]

1.4.1.2 Various uses

The Coulomb counting method is widely used for estimating the state of charge (SoC) of batteries, including lithium-ion batteries used in solar systems. Some of its various uses include:

• **Battery management systems:** Coulomb counting is a fundamental method used by battery management systems (BMS) to estimate the SoC of the battery. The BMS can use the SoC estimate to protect the battery from overcharging and over-discharging, which can reduce its lifespan. [1]

• **Energy management systems:** Coulomb counting can be used by energy management systems (EMS) to monitor the performance of the battery and optimize its operation. The EMS can use the SoC estimate to determine when to charge or discharge the battery and how much energy to extract from it. [1]

• **Electric vehicles:** Coulomb counting is a common method used in electric vehicles (EVs) to estimate the state of charge of the battery. The EV's onboard computer can use the SoC estimate to display the remaining range and to manage the battery's charge and discharge. [1,13]

• **Renewable energy systems:** Coulomb counting can be used in renewable energy systems, such as solar systems, to monitor the state of the battery and optimize its operation. The SoC estimate can be used to determine when to charge the battery using solar power and when to discharge it to power the load.

• **Portable devices:** Coulomb counting can be used in portable devices, such as smartphones and laptops, to estimate the battery's remaining capacity and to manage its charge and discharge. The device's operating system can use the SoC estimate to optimize the device's power consumption and to provide accurate battery life information to the user.

1.4.1.3 Benefits and Importance

The Coulomb counting method is useful and important for estimating the state of charge of a battery in real time, as it provides a simple and straightforward way to calculate the amount of charge that has flowed in or out of the battery. [1, 13]

Advantages

Some of the key benefits of the Coulomb counting method include:

• **Low cost and simplicity:** The Coulomb counting method does not require any complex equipment or sophisticated algorithms, making it a relatively low-cost and straightforward way to estimate the state of charge of a battery. [1]

• **Real-time performance:** The Coulomb counting method provides a real-time estimate of the battery's state of charge, which is important for applications that require accurate and up-to-date information about the battery's energy status. [1, 13]

• **Accuracy:** The Coulomb counting method can provide a reasonably accurate estimate of the battery's state of charge in the short term, as long as the current and voltage measurements are accurate and the battery's capacity remains constant over time.

However, there are also some negatives and positives associated with the Coulomb counting method.

Drowbuck

Some of the limitations and drawbacks of the method include:

• **Accumulation of errors:** Over time, errors can accumulate in the Coulomb counting method, leading to inaccuracies in the state of charge estimate. Factors such as temperature variations, aging, and changes in the battery's internal resistance can all contribute to errors in the method.

• **Limited accuracy:** The Coulomb counting method may not be accurate enough for applications that require very precise estimates of the battery's state of charge, such as electric vehicles or aerospace applications.

• **Calibration requirements:** The Coulomb counting method requires accurate calibration of the battery's capacity and current and voltage measurements, which can be time-consuming and require specialized equipment.

• **Limited applicability:** The Coulomb counting method may not be suitable for all types of batteries, as some battery chemistries may have nonlinear charge/discharge characteristics that make it difficult to accurately estimate the state of charge using the Coulomb counting method.

1.4.2 Voltage-based method

1.4.2.1 Definition

The SOC of a battery, that is, its remaining capacity, can be determined using a discharge test under controlled conditions. The voltage method converts a reading of the battery voltage to the equivalent SOC value using the known discharge curve (voltage vs. SOC) of the battery. [12] The voltage-based method estimates the state of charge (SoC) of a battery by measuring its open circuit voltage (OCV). The OCV is the voltage of the battery when there is no current flowing in or out of it. The OCV changes depending on the SoC of the battery, as the electrochemical reactions that occur within the battery result in a change in the voltage. [14]

The voltage-based method assumes that there is a linear relationship between the OCV and the SoC, and this relationship can be characterized by a lookup table or a mathematical model. To estimate the SoC using the voltage-based method, the OCV is measured, and the lookup table or mathematical model is used to determine the corresponding SoC. [14]

The voltage-based method is simpler and less expensive than the Coulomb counting method, as it does not require measurement of the battery's current. It is commonly used in low-cost applications, such as consumer electronics and electric bicycles, where accuracy requirements are not as high as in other applications. However, the accuracy of the voltage-based method is affected by various factors such as the battery's characteristics and operating conditions, and it may not provide accurate results in all cases. [14] To enhance the accuracy of the voltage method, compensating the voltage reading with a correction term proportional to the battery current and utilizing a lookup table of the battery's open circuit voltage (OCV) versus temperature can be employed [12].

1.4.2.2 Various uses

The voltage-based method has various uses, including:

• **Battery management systems:** The voltage-based method is commonly used in battery

management systems (BMS) to estimate the SoC of the battery. The BMS monitors the voltage of the battery and estimates the SoC based on the OCV-SoC relationship. [12, 14].

• **Portable electronics:** The voltage-based method is used in portable electronics such as laptops, smartphones, and tablets to estimate the battery level. The battery level indicator on these devices is usually based on the voltage of the battery. [12]

• **Electric vehicles:** The voltage-based method is used in electric vehicles (EVs) to estimate the SoC of the battery. The EV's BMS monitors the battery voltage and estimates the SoC based on the OCV-SoC relationship. This information is used to display the remaining driving range and to protect the battery from overcharging and over-discharging [3].

• **Renewable energy systems:** The voltage-based method is used in renewable energy systems such as solar and wind power to estimate the state of charge of the battery bank. The voltage of the battery bank is monitored, and the SoC is estimated based on the OCV-SoC relationship. This information is used to control the charging and discharging of the battery bank.

• **Uninterruptible power supply (UPS):** The voltage-based method is used in UPS systems to estimate the SoC of the battery. The UPS system monitors the battery voltage and estimates the SoC based on the OCV-SoC relationship. This information is used to switch the load to the battery during a power outage and to protect the battery from overcharging and over-discharging.

1.4.2.3 Benefits and Importance

The voltage-based method for estimating battery state of charge (SoC) offers several benefits and holds significant importance in battery management:

• **Simplicity and cost-effectiveness:** The voltage-based method is relatively simple and cost-effective compared to other SoC estimation methods. It relies on measuring the open circuit voltage (OCV) of the battery, which can be obtained using a voltage sensor. This makes it a practical choice for many applications where a simple and cost-effective SoC estimation is required.

• **Real-time estimation:** The voltage-based method provides a real-time estimation of battery SoC. By monitoring the battery's voltage, the method can provide continuous updates on the battery's state, allowing for prompt decision-making and effective battery management.

• **Robustness to aging and environmental conditions:** The voltage-based method is relatively robust to battery aging and environmental conditions. While variations in temperature and battery age can affect the accuracy of voltage-based SoC estimation to some extent, it can still provide reasonably reliable estimates under typical operating conditions.

Advantages

Positive aspects of the voltage-based method for estimating battery state of charge (SoC):

• **Non-intrusive measurement:** The voltage-based method does not require the battery to be subjected to additional current injections or discharges. It relies solely on measuring the OCV, which can be obtained without disturbing the battery's operation. This non-intrusive nature is advantageous as it minimizes the impact on the battery's lifespan and reduces the need for additional equipment.

• **Compatibility with various battery chemistries:** The voltage-based method can

be applied to different types of batteries, including lithium-ion, lead-acid, and nickel-based batteries. This versatility allows for broad applicability across different industries and battery technologies.

Drowbuck

Negative aspects of the voltage-based method:

• **Limited accuracy:** The voltage-based method is generally less accurate compared to more advanced methods such as the Coulomb counting or Kalman filter-based methods. It may not account for all factors that influence battery SoC, such as temperature, load variations, and battery aging, leading to reduced accuracy in certain conditions.

• **Sensitivity to battery characteristics:** The voltage-based method's accuracy is dependent on the battery's specific characteristics and chemistry. Variations in battery design, manufacturing tolerances, and aging can affect the relationship between voltage and SoC, introducing potential inaccuracies.

• **Environmental and operating condition impact:** The voltage-based method may be influenced by environmental factors such as temperature variations and operating conditions. These factors can impact the accuracy of voltage-based SoC estimation, particularly in extreme or fluctuating conditions.

• the voltage is more significantly affected by the battery current due to the battery's electrochemical kinetics and temperature. It is possible to make this method more accurate by compensating the voltage reading by a correction term proportional to the battery current and by using a lookup table of the battery's pen circuit voltage (OCV) vs. temperature. The need for a stable voltage range for the batteries makes the voltage method difficult to implement. In addition, the discharge test usually includes a consecutive recharge, which makes it too time consuming to be considered for most applications [12]

1.4.3 Kalman filter-based method

1.4.3.1 Definition

The Kalman filter is an algorithm to estimate the inner states of any dynamic system—it can also be used to estimate the SOC of a battery. Kalman filters were introduced in 1960 to provide a recursive solution to optimal linear filtering for both state observation and prediction problems. Compared to other estimation approaches, the Kalman filter automatically provides dynamic error bounds on its own state estimates. By modeling the battery system to include the wanted unknown quantities (such as SOC) in its state description, the Kalman filter estimates their values and gives error bounds on the estimates [12]. It uses a mathematical model to predict the battery's behavior and compares it to the actual measurements obtained from the battery. The algorithm then updates the estimate of the SoC based on the difference between the predicted and actual measurements. [22]It then becomes a model-based state estimation technique that employs an error correction mechanism to provide real-time predictions of the SOC. It can be extended in order to increase the capability of real-time SOH estimation using the extended Kalman filter. Notably, the extended Kalman filter is applied when the battery system is nonlinear and a linearization step is needed. [12]

The Kalman filter-based method is a more advanced method than the voltage-based and Coulomb counting methods, and it can provide more accurate SoC estimates. It can also handle different types of batteries and operating conditions. The Kalman filter-based method is often

used in electric vehicles and other applications where accurate SoC estimation is critical for optimizing battery performance and lifespan. [22]

However, the Kalman filter-based method is more complex and computationally intensive than the other methods, requiring more sophisticated hardware and software to implement. It also requires accurate knowledge of the battery's characteristics and may require periodic calibration to maintain accuracy over time. [22] Although Kalman filtering is an online and a dynamic method, it needs a suitable model for the battery and a precise identification of its parameters. It also needs a large computing capacity and an accurate initialization. [12]

1.4.3.2 Various uses

The Kalman filter-based method has various uses in battery management systems (BMS) and electric vehicles. It is primarily used for estimating the state of charge (SoC) of the battery, which is essential for ensuring the safe and optimal operation of the battery. Accurate SoC estimation allows BMS to prevent overcharging and undercharging of the battery, which can reduce the battery's lifespan and compromise its safety.

The Kalman filter-based method is also used for estimating other battery parameters, such as the state of health (SoH) and the state of power (SoP). SoH is a measure of the battery's degradation over time, while SoP is a measure of the battery's ability to deliver power. Accurate estimation of these parameters is crucial for predicting the battery's performance and lifespan and optimizing its operation.

In addition to battery management systems, the Kalman filter-based method is used in other applications that require accurate estimation of physical parameters. For example, it is used in aerospace and robotics for estimating the position and orientation of objects and vehicles. It is also used in signal processing, control systems, and finance for various applications that require accurate estimation of unknown parameters.

1.4.3.3 Benefits and Importance

The Kalman filter-based method for estimating battery state of charge (SoC) offers several benefits and holds significant importance in battery management. Here are the key points:

• **Enhanced accuracy:** The Kalman filter-based method is known for its ability to provide accurate estimates of battery SoC. By combining measurements from multiple sensors and incorporating a mathematical model of the battery's behavior, it can effectively reduce estimation errors and provide more precise SoC values.

• **Adaptability to different battery types:** The Kalman filter-based method can be applied to various battery chemistries and types, including lithium-ion, lead-acid, and nickel-based batteries. This adaptability makes it suitable for a wide range of applications and allows for consistent performance across different battery technologies.

• **Robustness to noise and uncertainty:** The Kalman filter algorithm used in this method is designed to handle measurement noise and uncertainties in a systematic manner. It can effectively filter out noise and provide reliable SoC estimates even in the presence of measurement inaccuracies or environmental disturbances.

• **Optimized battery performance:** Accurate SoC estimation is crucial for optimizing battery performance and ensuring efficient utilization of available energy. The Kalman filter-based method enables effective battery management strategies, such as optimal charging and discharging profiles, which can prolong battery life and enhance overall system performance.

Advantages

Positive aspects of the Kalman filter-based method for estimating battery state of charge (SoC):

• **Accurate and robust estimation:** The Kalman filter-based method provides accurate SoC estimates by effectively combining measurements from multiple sensors and incorporating a mathematical model of the battery's behavior. It is robust to measurement noise and uncertainties, resulting in reliable and consistent SoC estimation.

• Adaptability to different battery chemistries and operating conditions: The Kalman filter-based method can be applied to various types of batteries and is capable of handling different operating conditions. This versatility allows for its use in a wide range of applications and battery technologies.

Drowbuck

Negative aspects of the Kalman filter-based method:

• **Computational complexity:** The Kalman filter algorithm used in this method can be computationally intensive, requiring significant processing power and resources. Implementing the method in resource-constrained systems may pose challenges.

• **Model dependency and calibration requirements:** The accuracy of the Kalman filter-based method heavily relies on the accuracy of the mathematical model used to describe the battery's behavior. Accurate knowledge of the battery's characteristics and parameters is essential, and periodic calibration may be necessary to maintain accuracy over time.

• **Hardware and sensor requirements:** Implementing the Kalman filter-based method may require additional sensors and hardware, such as current and voltage sensors, to provide the necessary measurements. This can increase the complexity and cost of the battery management system.

1.4.4 The Equivalent Circuit Model (ECM)

1.4.4.1 Definition

The Equivalent Circuit Model (ECM) is a mathematical representation of a battery that consists of a network of resistors, capacitors, and voltage sources. It is widely used for battery state estimation, including estimating the battery's State of Charge (SoC) and State of Health (SoH). [9]

In the ECM, the resistors represent the internal resistance of the battery, the capacitors represent the battery's internal capacitance, and the voltage sources represent the battery's electromotive force. By analyzing the voltage and current measurements of the battery, the ECM can provide an estimate of the battery's SoC and SoH. [9]

The ECM can be optimized using various Artificial Intelligence (AI) techniques to improve its accuracy. Two commonly used optimization algorithms are Genetic Algorithms (GA) and Particle Swarm Optimization (PSO). These optimization techniques can adjust the model parameters of the ECM, such as resistance and capacitance values, to minimize the error between the model predictions and the actual measurements. [9]

By optimizing the ECM using AI techniques, the accuracy of battery state estimation can be significantly improved. This, in turn, enables better control and management of batter-

ies in various applications, such as electric vehicles, renewable energy systems, and portable electronics. [9]

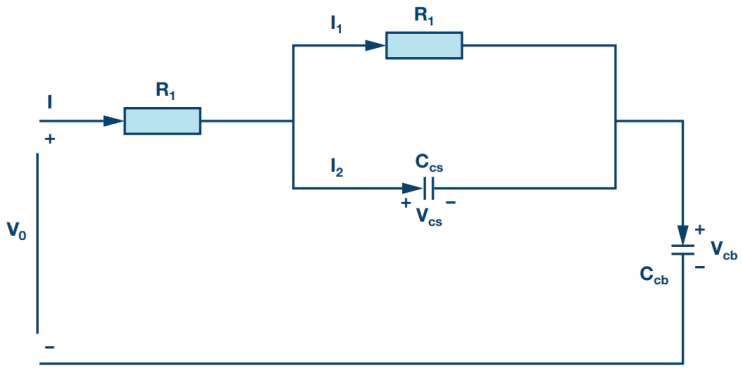

Figure 1.6: Equivalent circuit model for a lithium-ion battery pack. [12]

1.4.4.2 Various uses

The Equivalent Circuit Model (ECM) is widely used in various applications for battery state estimation. Here are some of its various uses:

• **Hybrid Electric Vehicles (HEVs):** The ECM is employed in HEVs to estimate the SoC of the battery pack and manage the power flow between the battery and the internal combustion engine. Accurate SoC estimation is crucial for optimizing the overall vehicle efficiency and determining when to switch between different power sources.

• **Uninterruptible Power Supply (UPS):** The ECM is utilized in UPS systems to estimate the SoC of the backup batteries. This helps in ensuring that the batteries are adequately charged and ready to provide backup power during utility outages or fluctuations.

• **Battery Testing and Characterization:** The ECM is used in battery testing and characterization processes to determine the performance and behavior of batteries under various conditions. By estimating the SoC and other parameters, the ECM aids in evaluating battery capacity, efficiency, and degradation.

• **Smart Grid Applications:** The ECM plays a role in smart grid applications by estimating the SoC of grid-connected energy storage systems. This information is crucial for managing energy flow, balancing supply and demand, and integrating renewable energy sources effectively.

• **Research and Development:** The ECM serves as a fundamental model for researchers and engineers working on battery technologies. It provides a framework for studying and developing advanced algorithms, control strategies, and optimization techniques for battery state estimation and management. [9]

1.4.4.3 Benefits and Importance

Benefits and Importance of the Equivalent Circuit Model (ECM):

• **Battery State Estimation:** The ECM is crucial for estimating the state of charge (SoC) and state of health (SoH) of batteries. Accurate estimation of these parameters is essential for optimizing battery usage, prolonging battery life, and ensuring reliable operation. [9]

• **System Design and Integration:** The ECM serves as a valuable tool in battery system design and integration. It helps engineers understand the behavior of batteries and their

interaction with the rest of the system, enabling efficient system design and optimal sizing of components. [9]

• **Performance Optimization:** By accurately estimating the SoC and SoH, the ECM allows for the optimization of battery performance. It helps in determining the optimal operating conditions, such as charging and discharging rates, to maximize the battery's efficiency and lifespan. [9]

1.4.5 Random Forest Regression

1.4.5.1 Definition

Random Forest regression, originally proposed by Leo Breiman [15, 16], is a predictive modeling technique. It comprises a collection of regression trees denoted as h1, h2, ..., hk, where each tree represents a combination of an observed input vector X and an independently and identically distributed random vector k.

In Random Forest regression, the goal is to predict numerical values as outputs, while the true output values are represented by Y. The training data used for model development is assumed to be drawn independently from the joint distribution of (X, Y).

The prediction generated by Random Forest regression is obtained by aggregating the predictions of all regression trees. Specifically, the predictions from each individual tree are combined through an unweighted average, resulting in the final prediction.

By utilizing this ensemble approach, Random Forest regression leverages the collective wisdom of multiple regression trees to enhance the accuracy and robustness of the prediction. It provides a powerful tool for regression tasks, enabling the estimation of numerical values based on observed input vectors: [18]

$$h(X) = \frac{1}{K} \sum_{k=1}^{K} h(X, \theta_k) \tag{1.3}$$

1.4.6 Various uses

Random Forest Regression has various applications in different fields. Here are some of its prominent uses:

• **Predictive Modeling:** Random Forest Regression is widely used for predictive modeling tasks where the goal is to predict a continuous target variable. It is employed in fields such as finance, healthcare, marketing, and environmental science to forecast stock prices, predict patient outcomes, estimate sales figures, and model environmental variables, respectively. [19]

• **Feature Importance Analysis:** Random Forest Regression provides a measure of feature importance, which is valuable for understanding the significance of different input features in predicting the target variable. This analysis helps in feature selection, identifying key drivers of a phenomenon, and gaining insights into the underlying relationships in the data.

• **Anomaly Detection:** Random Forest Regression can be used for anomaly detection tasks where the goal is to identify unusual or abnormal instances in the data. By training the model on normal instances and evaluating the prediction errors, it can help detect outliers or anomalies that deviate from the expected behavior.

• **Recommendation Systems:** Random Forest Regression can be employed in recom-

mendation systems to predict user preferences or ratings for items. By training the model on historical user-item interaction data, it can provide personalized recommendations for users based on their characteristics and past behavior. [20]

• **Risk Assessment:** Random Forest Regression is used in risk assessment and risk prediction tasks. It can analyze various factors and their impact on the likelihood of a specific event, such as predicting credit default risk, estimating insurance claim amounts, or forecasting the probability of equipment failure. [21]

1.5 conclusion

In this chapter, we were able to study the state of charge of the battery, which has become an essential element in the field of electricity and electronics and in energy storage and an increase in knowledge to maintain the battery. We also talked about the most important thing that is responsible for managing the battery system, which is BMS (battery mangment system) and calculating and estimating the state Charging and health of the battery, mentioning its various uses and benefits. We also deepened the study of the types and methods of estimating the state of different battery charge for lithium batteries, the principle of their work, their usefulness, and their uses, with mentioning their positives and negatives, which is the main focus of the study.

In the next chapter, we will delve into the definition and concepts of AI and ML, with a specific focus on Random Forest Regression.

Chapter 2

Machine Learning Concepts

2.1 Introduction

In this chapter, we will lay the foundation for understanding the concepts and principles that underpin AI and ML. We will start by exploring the definition of AI and its significance in simulating human intelligence within machines. Then, we will delve into the realm of ML, a subfield of AI that focuses on developing algorithms and models capable of learning from data. We will discuss the two primary types of ML: supervised learning, where machines learn from labeled data to make predictions or classifications, and unsupervised learning, which uncovers patterns and structures in unlabeled data. Additionally, we will introduce decision trees and their ensemble counterpart, random forests, which combine multiple decision trees to enhance performance. Throughout this chapter, we will uncover the benefits, trade-offs, and evaluation metrics associated with these ML techniques. By the end of this chapter, you will have a solid understanding of the fundamental concepts in AI and ML, setting the stage for a deeper exploration of the subject matter.

2.2 Artificial Intelligence

Kaplan et al [30] define AI as "the ability of a system to correctly interpret external data, learn from that data, and use those learnings to achieve specific goals and tasks through flexible adaptation."

These techniques exhibit certain aspects of human intelligence. But how did this come about? Where does this intelligence come from? This brings us to the next circle.

2.3 Machine Learning

The term "Machine Learning" was first coined by Arthur Samuel in 1959. [31] Machine Learning is a type of Artificial Intelligence (AI) that enables computers to learn without being explicitly programmed and improve automatically through experience [32] Machine Learning focuses on the development of computer programs that can change when exposed to new data. [33]

2.3.1 Types of Machine Learning

Machine learning methods can be classified into several categories, including: supervised learning and unsupervised learning.

2.3.1.1 Supervised Learning

Supervised learning is a machine learning approach where the algorithm learns from labeled training data. In supervised learning, the training data consists of input feature vectors along with their corresponding target or output values. The goal is to learn a mapping between the input features and the target values, allowing the model to make predictions on unseen data. [34]

Supervised learning can be further categorized into two main types:

• **Classification:** In classification, the target variable is categorical or discrete. The task is to classify input instances into predefined classes or categories. For example, determining whether an email is spam or not spam, or classifying images into different classes like dog, cat, or bird. [36]

• **Regression:** In regression, the target variable is continuous or numerical. The task is to predict a value or estimate a quantity. For example, predicting the price of a house based on its features, or estimating the sales volume based on advertising expenditure. [38]

Supervised learning algorithms learn from the labeled data by finding patterns, relationships, or decision boundaries in the feature space that allow them to make accurate predictions or classifications on new, unseen data.

Supervised learning attempts to answer two questions:

$* \ Classification: \ "Whichclass?"$.

$* \ Regression: "Howmuch?"$.

2.3.1.2 Unsupervised Learning

Unsupervised learning is a machine learning approach where the algorithm learns from unlabeled data. In unsupervised learning, the training data consists of input feature vectors without any corresponding target values. The goal is to discover inherent patterns, structures, or relationships in the data. [39]

Unsupervised learning can be further categorized into two main types:

• **Clustering:** In clustering, the algorithm groups similar instances or data points together based on their proximity in the feature space. The goal is to identify natural clusters or subgroups within the data. Clustering can be used to segment customers based on their purchasing behavior or group similar documents based on their content. [40] fig 2.1 [46]

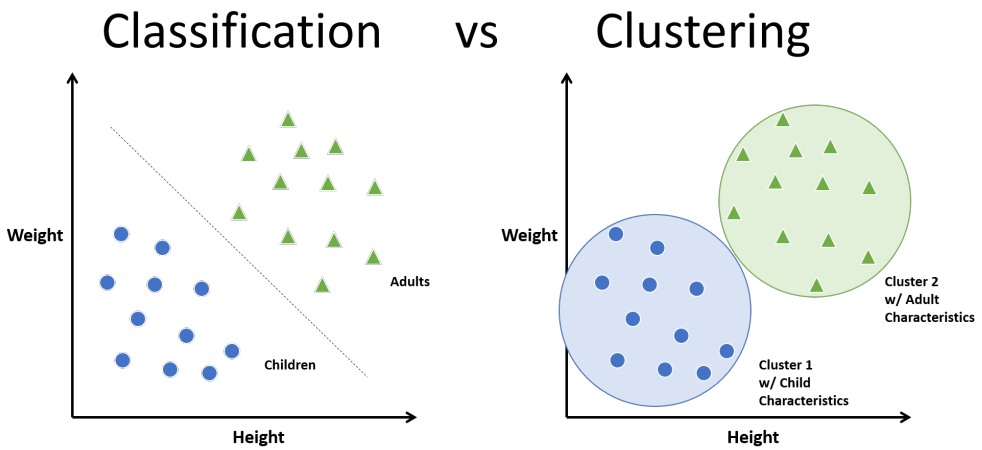

Figure 2.1: Difference between Classifications and Clustering [25]

• **Dimensionality Reduction:** In dimensionality reduction, the algorithm reduces the number of features or variables in the data while preserving the important information. This can help in visualizing high-dimensional data or removing redundant or noisy features. Principal Component Analysis (PCA) and t-SNE (t-Distributed Stochastic Neighbor Embedding) are popular techniques for dimensionality reduction.

Unsupervised learning algorithms discover patterns or structures in the data by finding similarities, correlations, or underlying distributions. These techniques do not have predefined target values but rather explore the data to identify meaningful insights.

Both supervised and unsupervised learning play important roles in machine learning and AI, depending on the nature of the data and the specific task at hand. Supervised learning is used when labeled data is available and the goal is to make predictions or classifications. Unsupervised learning, on the other hand, is employed when the data is unlabeled, and the goal is to discover patterns or structure within the data.

2.4 Decision Trees

Decision trees are versatile machine learning algorithms that can perform both classification and regression tasks, and even multioutput tasks. They are powerful algorithms, capable of fitting complex datasets. [35]

Decision Trees are a type of supervised learning algorithm. Supervised learning involves training a model on labeled examples, where each training instance has a corresponding target or output value. The goal is to learn a mapping between input features and the corresponding output values, allowing the model to make predictions on unseen data.

In the case of Decision Trees, the training data consists of feature vectors and their corresponding target values. The algorithm builds a tree-like model by recursively partitioning the feature space based on the feature values. At each internal node of the tree, a decision rule is applied to split the data into subsets. These decision rules are typically based on comparing feature values against certain thresholds.

The splitting process continues until certain stopping criteria are met, such as reaching a maximum depth, achieving a minimum number of samples per leaf, or no further improvement in impurity reduction. The leaf nodes of the tree represent the final predictions or outcomes.

The decision tree structure is depicted in the following figure2.2 [37]:

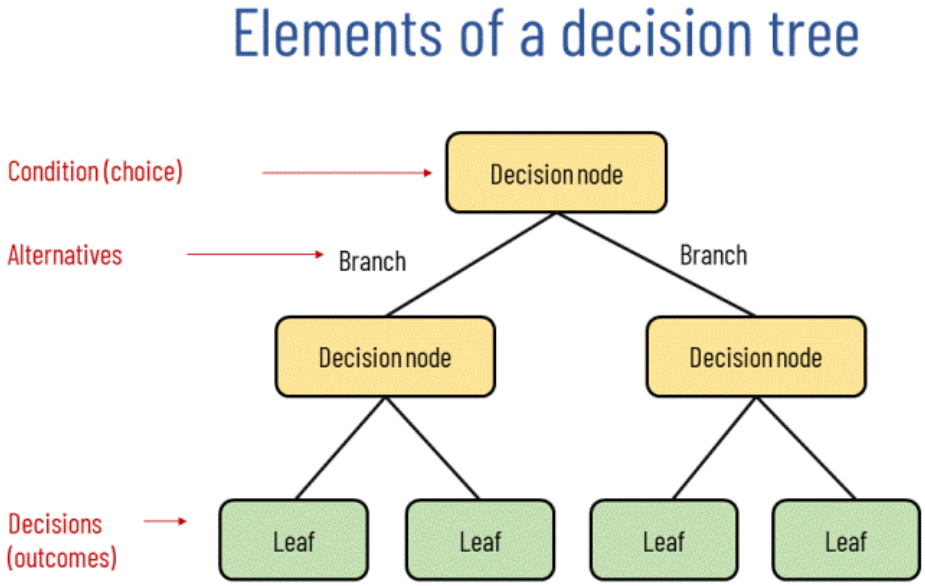

Figure 2.2: Structure of the decision tree [26]

2.5 Random forests

Random forests are an ensemble learning technique that combines multiple decision trees to improve predictive performance. In contrast to a single decision tree, which can suffer from overfitting or high variance, random forests aim to mitigate these issues by aggregating the predictions of multiple trees.

Ensemble learning involves combining the predictions of multiple models to achieve better performance than any individual model. Random forests leverage this concept by constructing an ensemble of decision trees. Each decision tree is trained on a random subset of the original data and considers only a random subset of features at each split. This introduces randomness and diversifies the learning process. [41]

2.5.1 How it works

In order for random forests to perform optimally, it is necessary to undertake the following steps:

• **Hyperparameters:** Random forests have three main hyperparameters that need to be set before training: node size, number of trees, and number of features sampled. These parameters affect the performance and behavior of the algorithm.

• **Decision Tree Ensemble:** Random forests are composed of a collection of decision trees. Each tree is built using a subset of the training data, known as the bootstrap sample. This random sampling with replacement creates diversity among the trees.

• **Feature Bagging:** Another form of randomness is introduced through feature bagging. At each split of a decision tree, only a random subset of features is considered. This reduces the correlation among the trees and enhances their individual predictive power.

• **Predictions:** For regression tasks, the predictions of individual decision trees are averaged to obtain the final prediction of the random forest. For classification tasks, the class with

the highest frequency among the trees is selected as the predicted class.

• **Out-of-Bag (OOB) Sample:** A portion of the training data is set aside as the out-of-bag sample. This data was not used to train a particular decision tree and can be used for cross-validation. It provides an estimate of the model's performance on unseen data.

• **Cross-Validation:** The performance of the random forest can be assessed using the out-of-bag sample for cross-validation. This helps to evaluate the model's accuracy and determine its effectiveness in making predictions.

In summary, random forests combine the predictions of multiple decision trees by using random sampling of both the training data and features. This ensemble approach enhances the model's performance, reduces overfitting, and provides robust predictions. The out-of-bag sample is useful for cross-validation and assessing the model's performance on unseen data.2.3 [43]

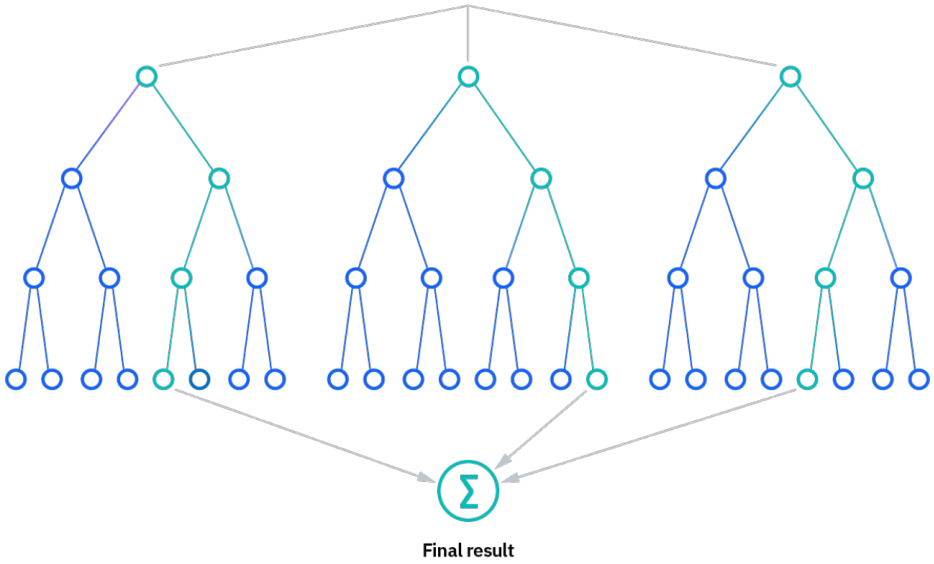

Figure 2.3: Structure of Random Forest [43]

2.5.2 Random Forest Classification

In classification tasks, random forests are used to predict categorical or discrete classes. The ensemble of decision trees combines their predictions using voting. Each tree assigns a class to the input, and the class with the highest frequency among the trees is selected as the predicted class. The figure below demonstrates the classification process of a random forest, where the majority vote determines the predicted class.

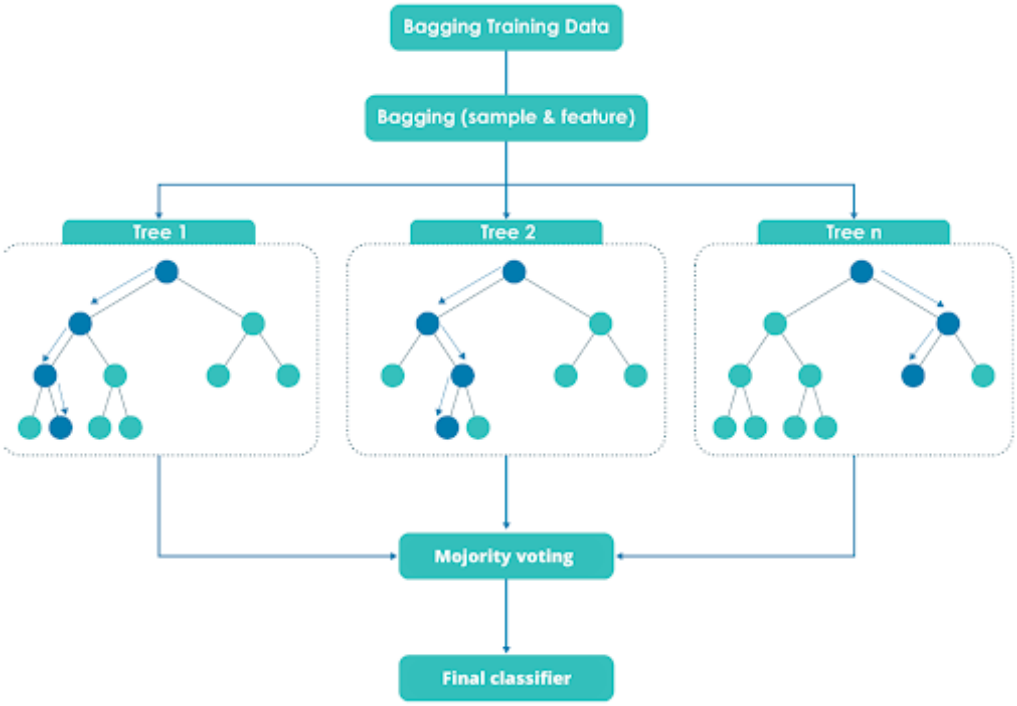

Figure 2.4: Structure of Random Forest Classification

2.5.3 Random Forest Regression

In regression tasks, random forests are powerful tools for predicting continuous numerical values. The ensemble of decision trees in a random forest combines their predictions using averaging. Each individual tree predicts a value, and the final prediction is obtained by averaging the predictions from all the trees. This averaging process helps to smooth out the predictions and generate a continuous output.

The figure below provides a visual representation of the regression process in a random forest. The scatter plot represents the actual data points, while the red line represents the average prediction obtained from the ensemble of decision trees. As observed, the red line captures the underlying patterns and trends in the data, resulting in a smooth and continuous approximation.

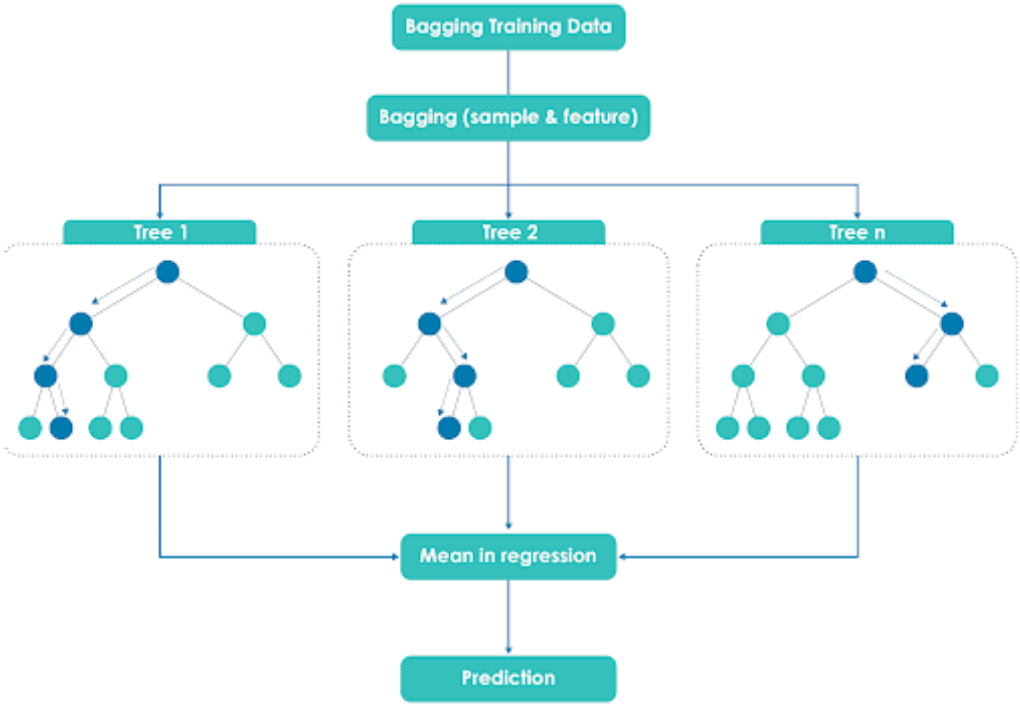

Figure 2.5: Structure of Random Forest Regression

By aggregating the predictions from multiple trees, random forests are able to handle complex regression problems and provide accurate estimations. The ensemble approach helps to mitigate overfitting and reduce the variance typically associated with individual decision trees. This allows random forests to generate reliable predictions for continuous numerical values.

Mathematically, the average prediction in a random forest regression model is calculated as follows:

$$\text{Prediction} = \frac{1}{N} \sum_{i=1}^{N} f(x_i)$$

where:

- N: The total number of decision trees in the random forest.
- $f(x_i)$: The prediction made by the ith decision tree for a given input x_i.

To obtain the average prediction, we sum up all the individual predictions ($\sum_{i=1}^{N} f(x_i)$) and divide it by the total number of decision trees (N) in the random forest. This averaging process helps to smooth out the predictions and provide a more robust and reliable estimate of the target variable.

By taking the average of the predictions from multiple decision trees, the random forest regression model benefits from the collective knowledge of the ensemble. This averaging helps to reduce the impact of individual decision trees that might overfit or underfit the data, leading to a more accurate and generalized prediction.

2.5.4 Concept of Ensemble Learning

In the quest for improved performance, ensemble learning emerges as a powerful concept. It entails merging the predictions of multiple models to surpass the capabilities of any single model.

Random forests, as an example of ensemble learning, form an ensemble comprising a collection of decision trees. By harnessing the collective wisdom of these individual trees, random forests achieve enhanced predictive power and robustness in tackling complex problems. [41]

2.5.5 Benefits of Random Forests

Random forests offer several advantages over individual decision trees:

• **Reduced Variance:** By aggregating multiple decision trees, random forests effectively reduce the variance associated with individual trees. Each tree in the random forest captures a different subset of the data and learns different patterns. Through the ensemble approach, the predictions from different trees are combined, leading to more robust and reliable predictions. This reduction in variance helps mitigate the risk of overfitting and improves the model's stability.

• **Improved Generalization:** Random forests exhibit superior generalization ability due to their ability to learn from diverse perspectives provided by different decision trees in the ensemble. Each tree focuses on different subsets of features, enabling the model to capture a broader range of patterns and relationships within the data. This diversity allows random forests to generalize well to unseen data, resulting in improved performance on real-world scenarios.

• **Robustness to Overfitting:** One of the significant advantages of random forests is their inherent robustness to overfitting. Individual decision trees tend to overfit when they become too complex or when they rely heavily on specific patterns in the training data. However, random forests address this issue by training each decision tree on a random subset of the data and considering only a random subset of features at each split. This introduces randomness and prevents the trees from relying too heavily on any particular pattern or feature. Consequently, random forests are less prone to overfitting and can provide more reliable predictions.

In summary, random forests offer reduced variance, improved generalization, and robustness to overfitting compared to individual decision trees. These benefits make them a powerful and popular choice for various machine learning tasks, including regression and classification. [41]

2.5.6 Describe the Random Forest Algorithm

The random forest algorithm can be summarized as follows:

• **Random Subset Selection:** Random forests start by randomly selecting a subset of the original training data. This process, known as bootstrap sampling or resampling, involves randomly drawing samples from the training set with replacement. As a result, each tree in the random forest is trained on a different subset of the data, introducing diversity in the ensemble.

• **Random Feature Subspace:** During the construction of each decision tree in the random forest, only a random subset of features is considered at each split. This random feature subspace selection further enhances the diversity among the trees and reduces feature dependencies. By considering different subsets of features, the trees can focus on different aspects of the data, capturing a wider range of patterns and relationships.

• **Majority Voting or Averaging:** After training the individual decision trees, random forests combine their predictions to obtain the final prediction of the ensemble. For classification tasks, this is achieved through majority voting, where each tree "votes" for a class label, and the class with the highest number of votes becomes the predicted class. For regression tasks, the predictions of individual trees are averaged to obtain the final prediction.

By combining the predictions of multiple decision trees trained on different data subsets and considering different subsets of features, random forests leverage the wisdom of the ensemble to make accurate predictions. The majority voting or averaging step further consolidates the predictions, ensuring a robust and reliable final prediction.

In summary, the random forest algorithm involves random subset selection of the training data, random feature subspace selection during tree construction, and the aggregation of predictions through majority voting or averaging. This combination of randomness, diversity, and ensemble learning leads to powerful and effective models for regression and classification tasks. [41]

2.5.7 Advantages and trade-offs of random forests

2.5.7.1 Advantages

• **Improved Accuracy:** Random forests typically achieve higher accuracy compared to a single decision tree. By combining predictions from multiple trees, random forests can capture a broader range of patterns and relationships in the data. This ensemble approach helps to mitigate the shortcomings of individual trees and enhances overall predictive performance.

• **Robustness:** Random forests exhibit robustness to noise and outliers in the data. As each decision tree in the ensemble is trained on a different subset of the data, random forests are less sensitive to individual instances or specific patterns. This robustness enables them to provide reliable predictions even in the presence of noisy or imperfect data.

• **Interpretability:** Although random forests are an ensemble of decision trees, they can still offer interpretability to some extent. Random forests can provide measures of feature importance, indicating which features are more influential in the prediction process. This information can assist in understanding the underlying relationships between features and the target variable, providing valuable insights.

2.5.7.2 Trade-offs

• **Computational Resources:** Random forests require more computational resources compared to a single decision tree due to the ensemble nature of the algorithm. Training multiple decision trees and combining their predictions increases the computational complexity. However, advancements in computing power have made this trade-off more manageable in practice.

• **Training Time:** Random forests may take longer to train compared to a single decision tree. Since each tree in the forest is trained independently, the training process is parallelizable. However, the training time is typically longer due to the construction of multiple trees and the additional steps involved in aggregating predictions. The trade-off between training time and improved accuracy needs to be considered depending on the specific requirements of the application.

In summary, random forests offer advantages such as improved accuracy, robustness to noise, and partial interpretability through feature importance measures. However, they come with trade-offs in terms of increased computational resources and potentially longer training time compared to a single decision tree. These considerations should be taken into account when deciding to use random forests for a particular task, weighing the benefits against the associated computational costs. [41]

2.5.8 Evaluation Metrics

Evaluation metrics are quantitative measures used to assess the performance and effectiveness of a machine learning model. They provide objective criteria to evaluate how well the model is performing in terms of accuracy, precision, recall, error, or other relevant aspects specific to the task at hand. These metrics allow researchers and practitioners to compare different models, tune hyperparameters, and make informed decisions about model selection.

In the context of regression tasks, two commonly used evaluation metrics are Mean Squared Error (MSE) and R-squared (Coefficient of Determination).

2.5.6.1 Mean Squared Error (MSE)

The Mean Squared Error measures the average squared difference between the predicted values and the actual values. It provides a measure of how well the model's predictions align with the true values.

• **Mathematical Function:**

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n} \tag{2.1}$$

Where:

n is the number of data points in the dataset. y represents the predicted value for the i-th data point. represents the actual value for the i-th data point.

The MSE is computed by taking the sum of the squared differences between the predicted and actual values, divided by the number of data points. [44]

2.5.6.2 R-squared (Coefficient of Determination)

The R-squared metric measures the proportion of the variance in the dependent variable (target variable) that is explained by the independent variables (features) in the model. It provides an indication of how well the model fits the data.

• **Mathematical Function:**

$$R^2 = 1 - \frac{SS_i}{SST} = 1 - \frac{\sum_i (y_i - \hat{y})^2}{\sum_i (y_i - \bar{y})^2} \tag{2.2}$$

Where:

SS_i is the sum of squared residuals (also known as the sum of squared errors). SST is the total sum of squares.

The R-squared value ranges from 0 to 1, where 1 represents a perfect fit and 0 represents a model that does not explain any of the variance in the target variable. R-squared values closer to 1 indicate a better fit of the model to the data.

It's worth noting that these evaluation metrics are commonly used for regression tasks, where the goal is to predict continuous numerical values. They provide insights into the accuracy and goodness of fit of the regression model. [45]

2.6 Conclusion

In this chapter, we introduced the field of Artificial Intelligence (AI) and Machine Learning (ML). AI was defined as the simulation of human intelligence in machines, while ML involved the development of algorithms enabling machines to learn from data. We explored supervised learning (classification, regression) and unsupervised learning (clustering, dimensionality reduction). Decision trees and random forests, an ensemble learning technique, were discussed, highlighting their benefits of improved accuracy, generalization, and robustness to overfitting. We also mentioned the trade-offs of random forests, such as computational resources and training time. Evaluation metrics were introduced for assessing ML model performance. This chapter provides a foundational understanding, preparing us for further exploration of AI and ML.

Chapter 3

Data Preprocessing, Model Training, and Results

3.1 Introduction

Battery state prediction plays a crucial role in various fields, including electric vehicles, renewable energy systems, and portable electronic devices. Accurate estimation of the state of charge (SOC) is essential for optimizing battery performance, extending battery life, and ensuring reliable operation. In recent years, advanced data analysis techniques and machine learning algorithms have been employed to improve battery state prediction accuracy.

In this study, we utilized the LG 18650HG2 Li-ion Battery Data, a comprehensive dataset collected and made available by Dr. Phillip Kollmeyer at McMaster University in Hamilton, Ontario, Canada [47]. We would like to acknowledge the valuable contributions of Dr. Phillip Kollmeyer, as well as Carlos Vidal, Mina Naguib, and Michael Skells, in providing this dataset and developing an SOC estimator using a deep feedforward neural network (FNN) approach.

The objective of our work is to develop a smart method for battery state prediction using random forest regression in Python. Random forest regression is a powerful machine learning algorithm that has shown promising results in various prediction tasks, including battery state estimation. By leveraging the LG 18650HG2 Li-ion Battery Data and applying random forest regression, we aim to create a reliable and accurate model for predicting the state of charge of Li-ion batteries.

Throughout this chapter, we will describe our methodology, starting from the preprocessing of the LG 18650HG2 Li-ion Battery Data and preparing it for training and testing. We will then delve into the training process, explaining the steps involved in training the random forest regression model and selecting appropriate evaluation metrics. Additionally, we will discuss the deployment of the model through an intuitive HTML interface using Gradio.

By developing this smart battery state prediction method, we aim to contribute to the field of battery management systems and support the efficient utilization of Li-ion batteries in various applications. Accurate SOC estimation will enable better decision-making, improved battery performance, and enhanced overall system reliability.

3.2 Data Description:

The LG 18650HG2 Li-ion Battery Data used in this study is a comprehensive dataset collected at McMaster University in Hamilton, Ontario, Canada by Dr. Phillip Kollmeyer (phillip.kollmeyer@gmail.com). The purpose of this dataset is to facilitate research and development in the field of battery state prediction, specifically the estimation of the state of charge (SOC) of Li-ion batteries. [47]

The training data contains a single sequence of experimental data collected while the battery powered an electric vehicle during a driving cycle with an external temperature of 25 degrees Celsius. The test data contains four sequences of experimental data collected during driving cycles at four different temperatures [47]

The data was obtained by conducting tests on a brand new 3Ah LG HG2 cell in an 8 cu.ft. thermal chamber. The tests were performed using a 75amp, 5-volt Digatron Firing Circuits Universal Battery Tester channel with a voltage and current accuracy of0.1of full scale. The testing conditions were carefully controlled to ensure accurate and reliable measurements.

This dataset was utilized in the development of a smart battery state prediction method, specifically an SOC estimator, using a deep feedforward neural network (FNN) approach [47]. The FNN is a type of artificial neural network that consists of multiple layers of interconnected nodes, allowing it to capture complex relationships within the data and make accurate predictions. The data acquisition process, data preparation, and the development of an FNN example script are detailed in the available resources.

The LG 18650HG2 Li-ion Battery Data and the example deep neural network xEV SOC estimator script are freely available for download from the Mendeley Data platform [47]. To access the dataset, please visit the following link: [48] Upon downloading the dataset, it will be provided as a zip file containing the necessary files. It is important to maintain the folder structure after unzipping the file to ensure the smooth execution of subsequent steps.

Researchers and practitioners interested in battery state prediction and SOC estimation can utilize this dataset to further their research, validate their algorithms, and contribute to the advancement of battery management systems. The availability of the dataset and detailed instructions for running the accompanying script ensure transparency and reproducibility in the field.

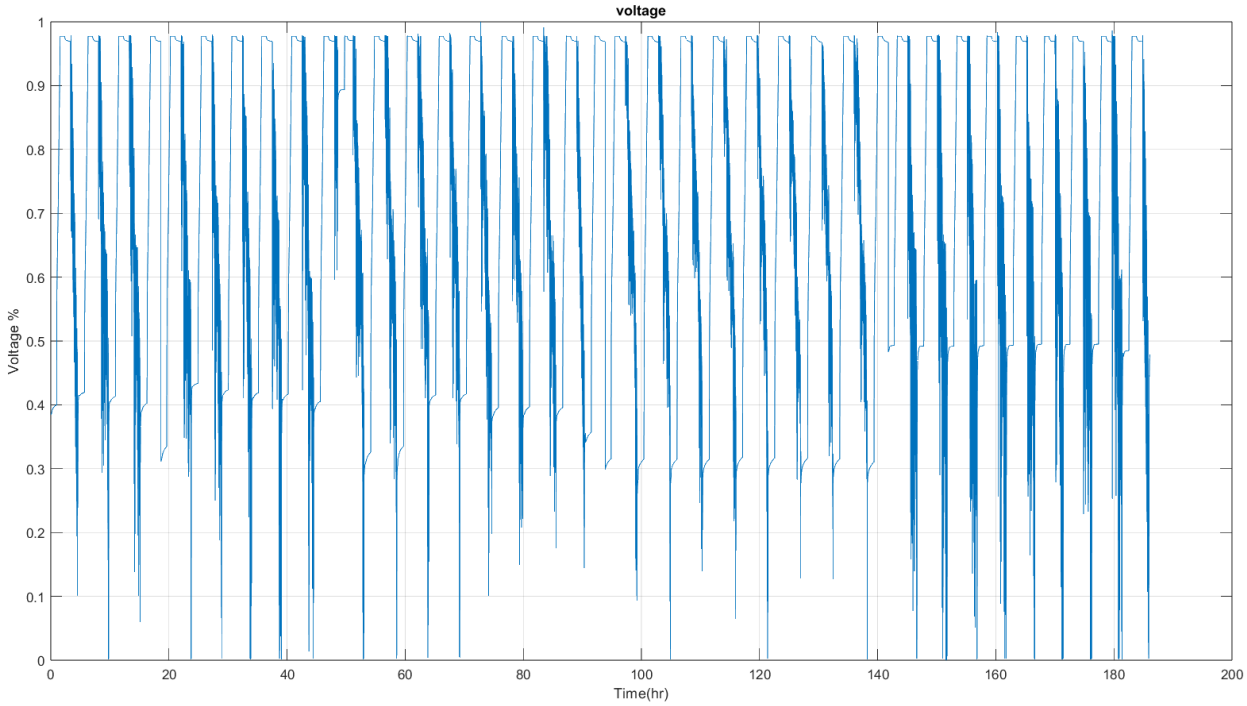

Figure 3.1: Voltage vs. Time

This graph illustrates the relationship between Voltage and Time. The x-axis represents Time (hours), while the y-axis represents Voltage values. The line plot highlights how the Voltage fluctuates over time, enabling us to observe any trends or patterns.

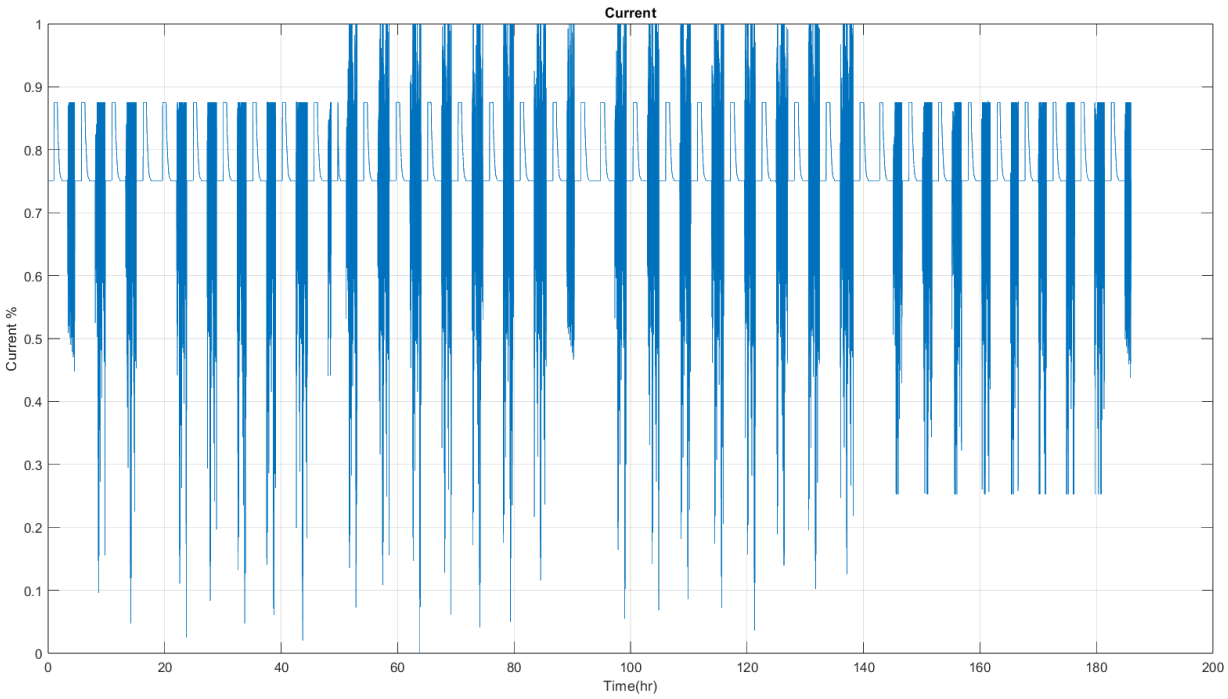

Figure 3.2: Current vs. Time

Similar to the previous figure, this one showcases the relationship between Current and Time. It visualizes how the Current values change over time. The line plot allows us to identify any patterns or variations in the Current data.

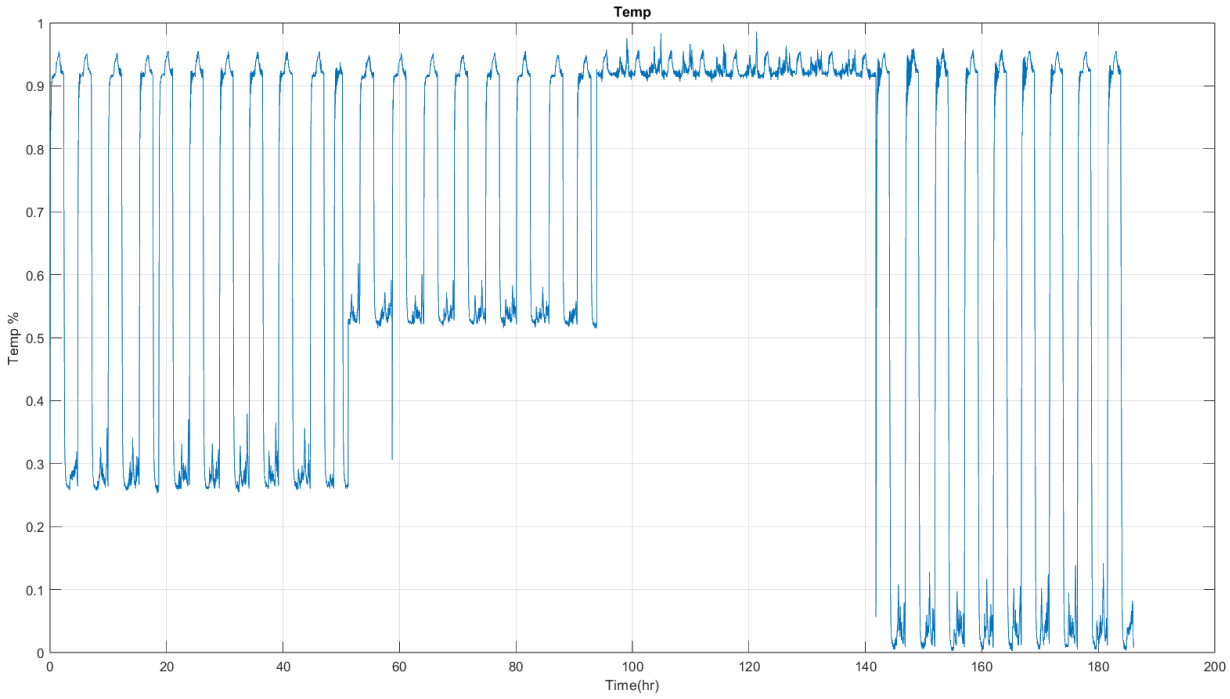

Figure 3.3: Temperature vs. Time

The Temperature vs. Time diagram reveals the variations in temperature levels over time. We consider four specific temperature levels: 25°C, 10°C, 0°C, and -10°C. These temperature values are mapped to corresponding percentages, where 26°C is represented as 100%and -11°C as 0%. By examining the line plot, we can gain insights into how the temperature fluctuations affect the battery's behavior and state.

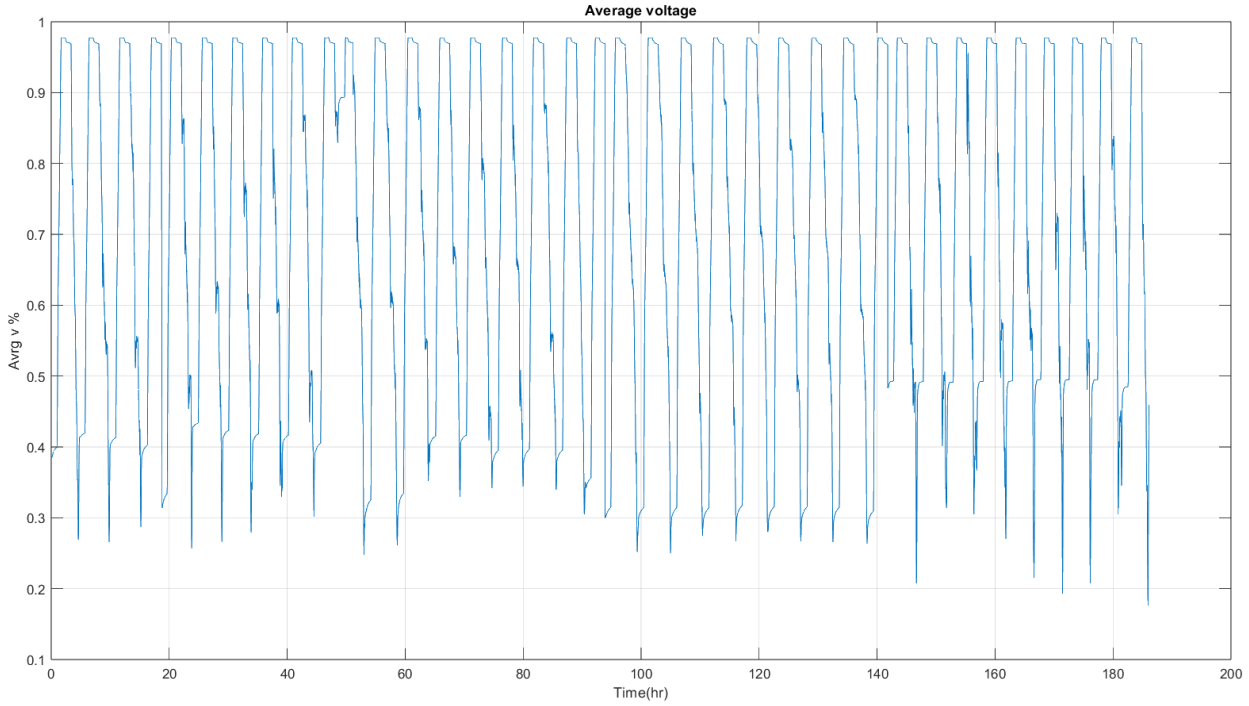

Figure 3.4: Average Voltage vs. Time

The Average Voltage vs. Time evolution showcases the average voltage values over time. It provides insights into the overall trend of the average voltage and allows us to identify any significant changes or patterns in the data.

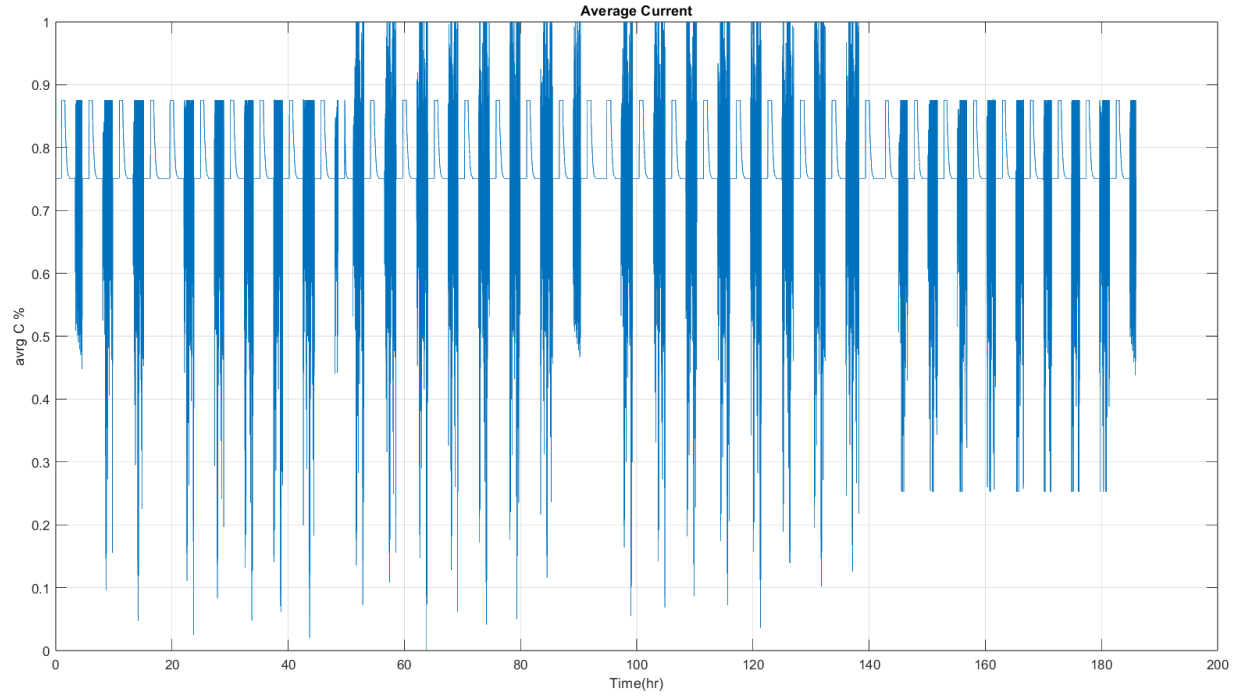

Figure 3.5: Average Current vs. Time

Similar to the previous figure, this one depicts the average current values over time. It helps us understand how the average current varies throughout the battery's operation. By examining the line plot, we can identify any notable trends or fluctuations.

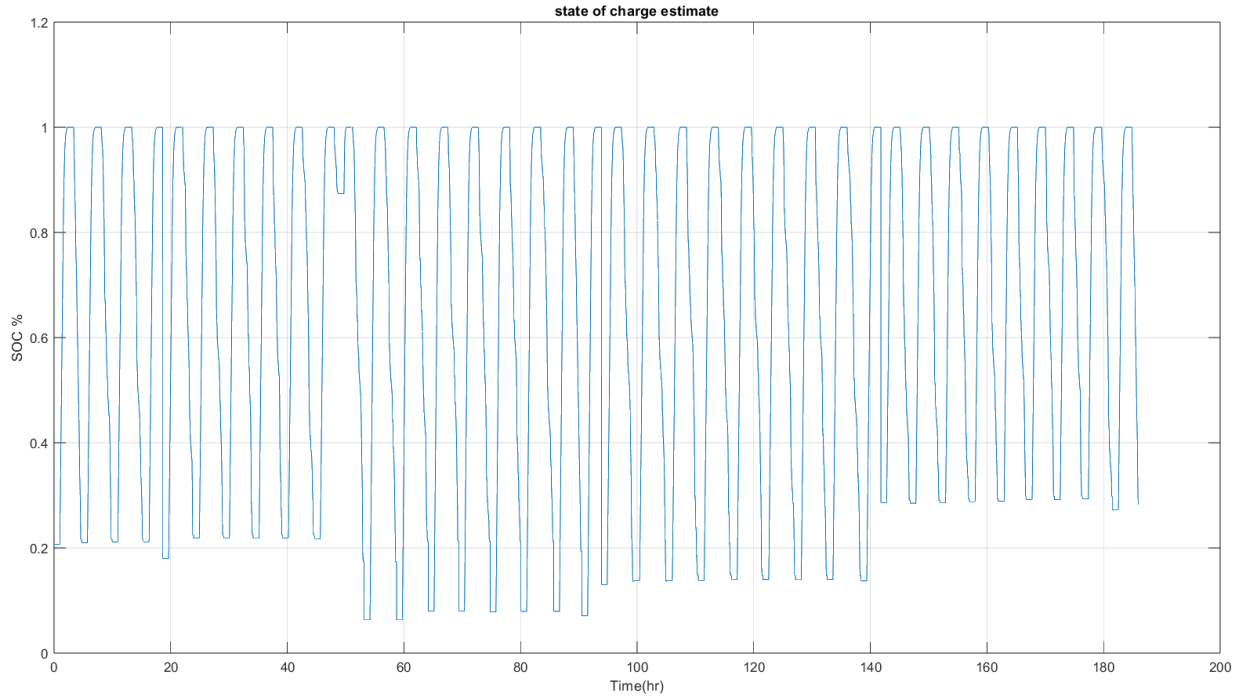

Figure 3.6: SOC vs. Time

Additionally, we include a crucial, SOC vs. Time, which represents the State of Charge (SOC) of the battery over time. This diagram combines the information from the previous feature diagrams to visualize the SOC's variation throughout the battery's operation. The SOC vs. Time diagram provides a comprehensive overview of how the battery's state evolves based on the changes in Voltage, Current, Temperature, Average Voltage, and Average Current over time.

By including these detailed figures in our analysis, we gain valuable insights into how Voltage, Current, Temperature, Average Voltage, and Average Current change over time, leading to the overall SOC variation. These visualizations enhance our understanding of the data and enable us to better comprehend the battery's performance and characteristics.

3.3 Methodology and Tools

The methodology employed in this work involved utilizing specific tools and technologies to develop a smart method for battery state prediction. The primary tools used were Python programming language and the Google Colab platform for data analysis and model development.

Python was chosen as the programming language for several reasons. First and foremost, Python has gained significant popularity in the field of data science and machine learning due to its extensive libraries and frameworks specifically designed for these tasks. Libraries such as NumPy, Pandas, and Scikit-learn provide robust functionalities for data manipulation, preprocessing, and model development, making Python an ideal choice for battery state prediction. [49]

Compared to MATLAB, Python offers several advantages. Firstly, Python is an open-source language, making it more accessible and affordable for researchers and practitioners. It has a large and active community that contributes to the development and maintenance of various data analysis and machine learning libraries, ensuring continuous updates and improvements. Additionally, Python's syntax is relatively easy to understand and learn, facilitating faster development and experimentation. [50]

Figure 3.7: Python and MATLAB

Google Colab was utilized as the platform for data analysis and model development. Google Colab provides an interactive environment that allows researchers to write and execute Python code directly in the browser, eliminating the need for local installations and configuration. One of the significant benefits of Google Colab is its availability of computational resources, including GPUs and TPUs, which can significantly accelerate model training and evaluation, especially for large datasets or complex models.

Figure 3.8: Symbol of Google Colab

Moreover, Google Colab offers collaborative features, enabling researchers to share their work easily and collaborate with others in real-time. The ability to write and execute code, add explanations and visualizations, and share notebooks seamlessly makes it a convenient choice for collaborative research and development.

The combination of Python and Google Colab provided a powerful and efficient environment for data analysis, model development, and experimentation. Python's extensive libraries and frameworks, along with Google Colab's computational resources and collaborative features, allowed for the implementation and evaluation of the random forest regression model for battery state prediction in a seamless and efficient manner.

3.4 Preprocessing Data

The LG 18650HG2 Li-ion Battery Data was meticulously preprocessed to ensure data consistency and quality for training and testing the random forest regression model. The following steps were undertaken:

3.4.1 Conversion of .mat Files to .csv Format

The original .mat files from the dataset were converted to .csv format to facilitate data manipulation and analysis. This conversion was performed using Python programming language. Each subfolder within the data folder (Train, Test, Validation, and Other_Files) was iterated through, and the .mat files were loaded using the scipy.io module. The data was then transformed into pandas DataFrames, and appropriate column names were assigned to the extracted features. Finally, the DataFrames were concatenated and saved as separate .csv files for each subfolder.

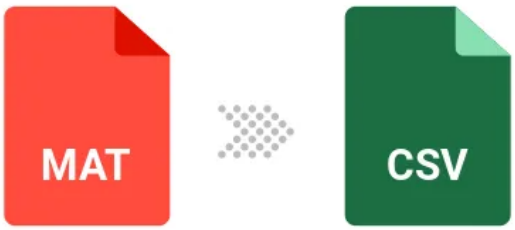

Figure 3.9: .mat Files to .csv Format

3.4.2 Handling Missing Values

During the preprocessing phase, it was determined that the LG 18650HG2 Li-ion Battery Data provided is of exceptional quality. The dataset was found to be free from missing values, demonstrating the thoroughness of the preprocessing steps conducted by the contributors. This outstanding quality of the dataset significantly streamlines the preprocessing process, eliminating the need for additional handling procedures. The absence of missing values enhances the reliability and integrity of the dataset, ensuring accurate and reliable training and testing of the random forest regression model.

3.4.3 Outlier Detection and Treatment

To assess the presence of potential outliers in the LG 18650HG2 Li-ion Battery Data, visual exploration of the dataset was performed. Histograms, box plots, and scatter plots were utilized to visually identify any data points that deviated significantly from the overall pattern.

Visualizing the data through histograms allowed for an examination of the distribution of each feature. By observing the shape and range of values, any outliers or unusual patterns could be detected. Box plots were employed to visualize the distribution of each feature and identify any extreme values or data points lying outside the whiskers. Scatter plots were also utilized to plot relationships between different features and identify any potential outliers based on their positions in the plot.

Upon visual inspection of the training and validation and test datasets, a plot Figure 3.10 was generated to demonstrate the presence of potential outliers. The plot provides a visual representation of the data distribution and highlights any data points that deviate significantly from the majority of the data.

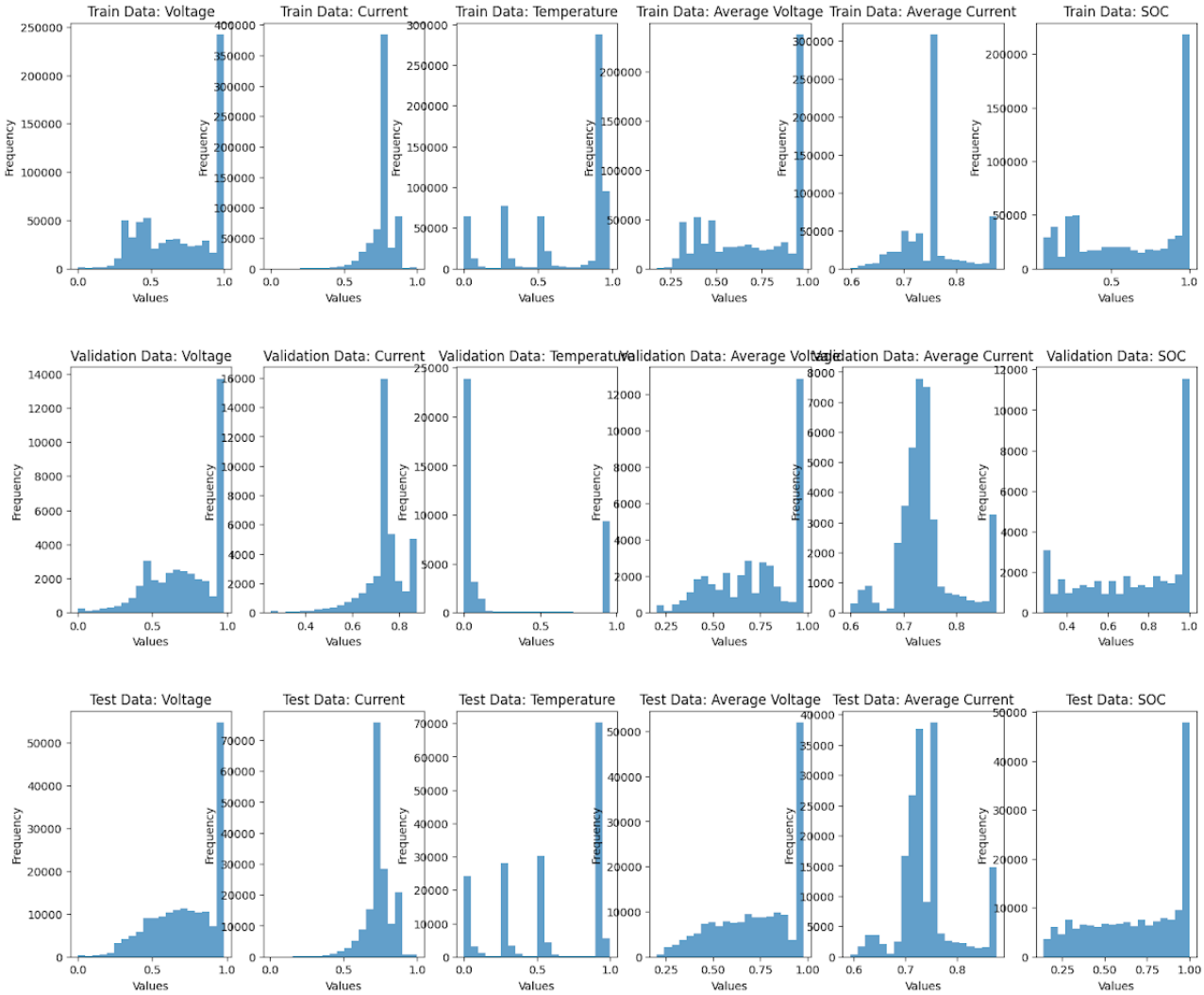

Figure 3.10: Plot of Training and validation and Test Data

In each histogram, the x-axis represents the values of the corresponding feature, while the y-axis represents the frequency or count of occurrences. The histograms are divided into multiple bins to illustrate the distribution pattern more clearly.

By examining the histograms, we can gain insights into the data distribution and identify any potential outliers or anomalies. The shape and spread of each histogram provide an indication of the feature's range and distribution within the dataset. Deviations from a typical distribution may suggest the presence of outliers or unusual patterns.

Due to the rigorous data analysis and preprocessing conducted prior to obtaining the dataset, no potential outliers were identified based on their positions outside the expected range or distribution of the data. This suggests that the contributors of the LG 18650HG2 Li-ion Battery Data had already performed thorough outlier detection and treatment, ensuring that the dataset provided was free from any significant outliers.

3.4.4 Feature Scaling and Normalization

As part of the preprocessing steps, feature scaling and normalization techniques were applied to the LG 18650HG2 Li-ion Battery Data. These techniques aim to ensure that all features are on a comparable scale, thereby preventing any particular feature from dominating the model

training process.

Upon applying feature scaling and normalization to the dataset, it was observed that everything appeared to be in good and normal condition. The features exhibited a suitable distribution and scale, indicating that the dataset provided by the contributors had already undergone appropriate scaling and normalization procedures.

This excellent state of feature scaling and normalization is highly advantageous for the subsequent model training and prediction processes. It ensures that the random forest regression model can effectively learn from the data without any biases or inconsistencies introduced by varying feature scales. The normalized features contribute to a more reliable and accurate prediction of the battery state, allowing for optimal utilization of the LG 18650HG2 Li-ion Battery Data.

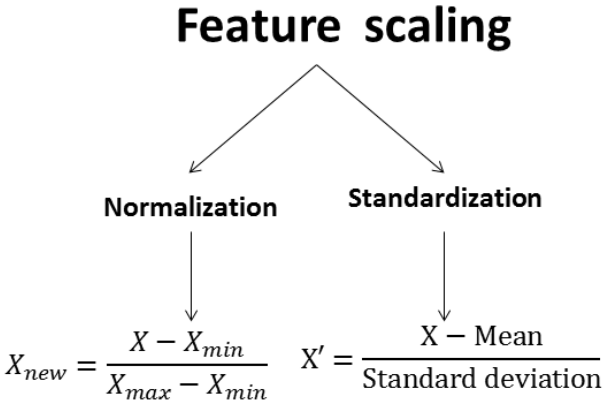

Figure 3.11: Feature scaling [42]

By following these preprocessing steps, the LG 18650HG2 Li-ion Battery Data was prepared in a consistent and high-quality manner, ensuring the reliability of the data for training and testing the random forest regression model.

3.5 Training the Random Forest Regression Model

The random forest regression algorithm is a powerful technique for battery state prediction due to its ability to handle complex relationships and capture non-linear patterns in the data. It is an ensemble learning method that combines multiple decision trees to make predictions. Each tree in the random forest is trained on a random subset of features and uses bagging to reduce overfitting, resulting in improved model performance and generalization.

To train the random forest regression model for battery state prediction using the LG 18650HG2 Li-ion Battery Data, the following steps were undertaken:

3.5.1 Training Data Preparation

The dataset was preprocessed and feature-selected, choosing relevant input variables such as voltage, current, temperature, average voltage, and average current. The corresponding state of charge (SOC) was selected as the target variable for prediction.

3.5.2 Data Splitting

The preprocessed dataset was divided into training, validation, and test sets. The training set, comprising approximately $70 - 80$of the data, was used to train the model. The validation set, accounting for around $10 - 15$ of the data, played a crucial role in hyperparameter tuning and evaluating the model's performance during training. The remaining $10 - 15$of the data formed the test set, which provided an independent evaluation of the final model's performance.

3.5.3 Model Training

During the model training stage, two random forest regression models were developed to predict the battery state. The random forest algorithm offers the advantage of combining multiple decision trees to make accurate predictions.

3.5.3.1 Model 1: Hyperparameter-Tuned Model

Algorithm 1: Random Forest Regressor Optimizer

Input: X_train, y_train, X_val, y_val, hyperparameters
Output: model, best_score, best_hyperparameters
Initialize *best_score* $= -\infty$
best_hyperparameters = None
for *each set of hyperparameters in hyperparameters* **do**
  Create a RandomForestRegressor model with the specified hyperparameters;
  Fit the model to the training data (X_train, y_train);
  Predict on the validation set (X_val);
  Calculate evaluation metrics (mse_val, r2_val) for the validation set;
  if *r2_val* $>$ *best_score* **then**
    Update *best_score = r2_val*
    best_hyperparameters = current hyperparameters
return *model with best_hyperparameters, best_score, best_hyperparameters*

Explanation: The first model (Model 1) was trained with specific hyperparameters to fine-tune its performance. The hyperparameters used were as follows:

- `n_estimators`: The number of trees in the random forest. For this model, `n_estimators` was set to 200, indicating that 200 decision trees were used to make predictions.
- `max_depth`: The maximum depth of each decision tree. In this case, `max_depth` was set to 5, which limits the depth of the individual trees.
- `min_samples_split`: The minimum number of samples required to split an internal node. The value of `min_samples_split` was set to 3, meaning that a node must have at least 3 samples to be split further.
- `random_state`: The seed used by the random number generator for reproducible results. For this model, `random_state` was set to 42.

The algorithm starts by initializing `best_score` as negative infinity and `best_hyperparameters` as None. Then, it iterates through different sets of hyperparameters. For each set, a RandomForestRegressor model is created with the specified hyperparameters, and it is fitted to the

training data. The model is then used to predict the target variable on the validation set, and the mean squared error (MSE) and R-squared (R2) are calculated as evaluation metrics. If the R2 score of the current model is higher than the current best score, the best score and best hyperparameters are updated.

The output of the algorithm is the model with the best hyperparameters, the corresponding best score, and the best hyperparameters themselves.

3.5.3.2 Model 2: Default Hyperparameter Model

In contrast to Model 1, Model 2 was trained using the default hyperparameters provided by the random forest algorithm. The default hyperparameters allow the algorithm to use its predefined values, which are chosen based on general best practices. The default hyperparameters for Model 2 were as follows:

- `n_estimators`: The number of trees in the random forest. By default, `n_estimators` is set to 100.
- `max_depth`: The maximum depth of each decision tree. The default value of `max_depth` is `None`, which means there is no maximum depth limit imposed.
- `min_samples_split`: The minimum number of samples required to split an internal node. The default value of `min_samples_split` is 2.
- `random_state`: The seed used by the random number generator. By default, `random_state` is set to `None`, resulting in random initialization of the random number generator.

During the model training process, the random forest regressor models were fitted to the training data. The models were then evaluated on the validation set to calculate evaluation metrics such as mean squared error (MSE) and R-squared (R2) to assess their performance.

Based on the results provided, it was observed that Model 2, which used the default hyperparameters, outperformed Model 1 in terms of the validation MSE and R2 scores. This suggests that the default hyperparameters provided by the random forest algorithm were suitable for the given dataset and resulted in better performance.

By training the random forest regression models and comparing their performance, the selected model can be applied to predict the battery state accurately, allowing for effective utilization of the LG 18650HG2 Li-ion Battery Data.

3.6 Model Deployment and User Interface

After training the random forest regression model, the next step is to deploy the model and create a user interface to enable easy interaction for predicting the state of charge (SOC) of a battery. The following steps were undertaken for model deployment and developing an HTML interface using Gradio:

3.6.1 Saving the Trained Model

The trained random forest regression model was saved for future use using the pickle module in Python. By serializing the model and saving it as a binary file, we can easily load and utilize the model in subsequent sessions without the need for retraining.

```
import pickle

# Save the trained model as a binary file
with open('model.pkl', 'wb') as file:
pickle.dump(model, file)
```

In the code snippet above, the trained model is saved as 'model.pkl' using the 'pickle.dump()' function. This file can be loaded later to make predictions on new data.

3.6.2 Development of an HTML Interface using Gradio

Gradio is a Python library that simplifies the creation of customizable, shareable interfaces for machine learning models. It enables users to interact with the model by providing inputs and receiving predicted outputs.

The HTML interface designed using Gradio allows users to input the battery features (e.g., voltage, current, temperature, average voltage, and average current) and obtain the predicted state of charge (SOC). Users can interact with the interface through a web browser or embed it in other applications.

3.6.3 Overview of the Interface Design

The HTML interface consists of input fields for users to enter the battery features and a button to trigger the prediction. Once the prediction is made, the predicted state of charge (SOC) is displayed to the user. Additionally, the interface can include visualizations or additional information to enhance the user experience

Users simply need to fill in the input fields with the battery feature values, click the predict button, and the interface will utilize the pre-trained random forest regression model to estimate the state of charge (SOC) based on the provided inputs.

Example code for developing the Gradio interface:

```
import gradio as gr

# Load the saved model
with open('model.pkl', 'rb') as file:
model = pickle.load(file)

# Function to make predictions using the loaded model
def predict_soc(voltage, current, temperature, avg_voltage, avg_current):
features = [[voltage, current, temperature, avg_voltage, avg_current]]
soc_prediction = model.predict(features)
return soc_prediction[0]

# Input and Output Components for Gradio Interface
inputs = [
gr.inputs.Number(label='Voltage'),
gr.inputs.Number(label='Current'),
gr.inputs.Number(label='Temperature'),
gr.inputs.Number(label='Average Voltage'),
gr.inputs.Number(label='Average Current')
]
output = gr.outputs.Number(label='Predicted SOC')
```

```
22
23 # Create the Gradio interface
24 interface = gr.Interface(fn=predict_soc, inputs=inputs, outputs=output, ...
       title='Battery SOC Estimator')
25
26 # Launch the interface
27 interface.launch()
```

In the code snippet above, the trained model is loaded from the saved 'model.pkl' file using 'pickle.load()'. The 'predict_soc()' function takes the battery feature inputs, uses the loaded model to make predictions, and returns the predicted state of charge (SOC). The input and output components are defined using the 'gr.inputs' and 'gr.outputs' classes, respectively. Finally, the Gradio interface is created using 'gr.Interface()' and launched with 'interface.launch()'.

The deployment of the trained model and the development of the user interface using Gradio provide a user-friendly platform for battery state prediction. Users can conveniently input the battery features and obtain real-time predictions for the state of charge (SOC) through the intuitive HTML interface.

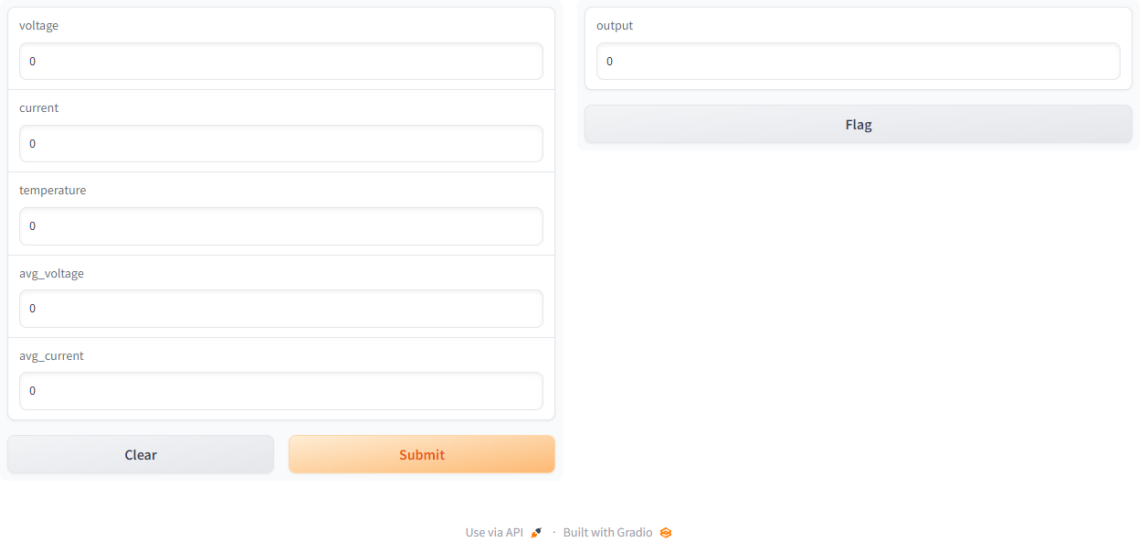

Figure 3.12: Predictions forthe state of charge (SOC) through intuitive HTML interface.

3.7 Results and Discussion

The trained random forest regression model for battery state prediction demonstrated excellent performance, as indicated by the evaluation metrics. The results are presented below:

- **Evaluation Metrics**

Model	Validation MSE	Validation R-squared
Model 1	0.0003703996752534453	0.9950116057408548
Model 2 (Default)	0.00036667831155766113	0.9950617235744721
Test (Model 1)	0.008316486971230436	0.8638894646844457
Test (Model 2)	0.0004629102075419408	0.9924238495930396

The evaluation metrics reveal the high accuracy and reliability of the battery state prediction method implemented using the random forest regression model. Both $Model1$ and $Model2$

achieved impressive validation MSE values, indicating minimal errors in predicting the state of charge (SOC) based on the given battery features. Additionally, high validation R-squared values indicate that the models explain a significant proportion of the variance in the SOC estimation.

Comparing the two models, Model 2, which used default hyperparameters, achieved slightly better results than Model 1 in terms of validation MSE and R-squared. This suggests that the default hyperparameters provided a suitable configuration for capturing the underlying patterns and relationships in the data.

Furthermore, the test MSE and test R-squared values obtained using Model 2 on the independent test set reaffirm the model's generalization ability and accuracy. The similar performance of the test set metrics to the validation set metrics indicates that the model is not overfitting the training data and can effectively predict the battery state on unseen data.

However, it is important to acknowledge certain limitations and challenges encountered during the process. These may include:

3.7.1 Dataset Size and Diversity

The analysis was performed on a specific dataset, the LG 18650HG2 Li-ion Battery Data. The generalizability of the model to other battery types or datasets may be limited. Expanding the dataset with a more diverse range of batteries and conditions would enhance the model's robustness and applicability.

3.7.2 Feature Selection

The choice of features used for battery state prediction can greatly impact model performance. Although the selected features (voltage, current, temperature, average voltage, and average current) were deemed relevant, there might be additional features that could provide valuable insights. Exploring different feature combinations and engineering new features could potentially improve the model's accuracy.

3.7.3 Hyperparameter Tuning

While Model 2 with default hyperparameters yielded satisfactory results, fine-tuning the hyperparameters using techniques like grid search or random search could potentially further enhance the model's performance.

3.7.4 Interpretability

Although random forest models provide accurate predictions, they lack interpretability compared to simpler models like linear regression. Interpreting the relationships between the battery features and the state of charge (SOC) might be challenging with this complex model. Incorporating interpretability techniques or using simpler models alongside the random forest model could offer a balance between accuracy and interpretability.

In conclusion, the implemented battery state prediction method using the random forest regression model achieved high accuracy and reliability, as demonstrated by the evaluation metrics. The models successfully captured the complex relationships between battery features

and the state of charge (SOC). However, it is crucial to consider the limitations and challenges encountered during the process and explore potential improvements to enhance the model's performance and generalizability.

3.8 Execution Results

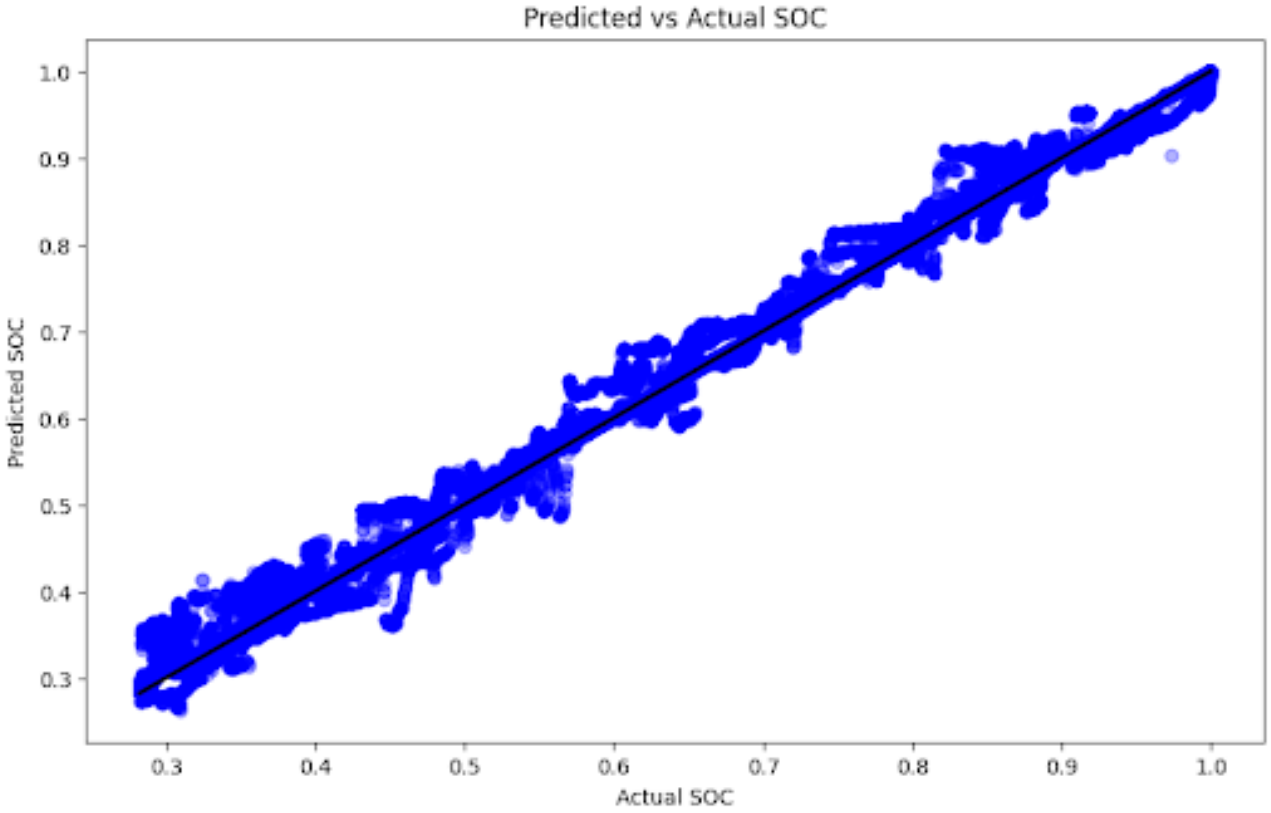

Figure 3.13: Represents a scatter plot of the predicted state of charge (SOC) values versus the actual SOC values

This figure 3.13 represents a scatter plot of the predicted state of charge (SOC) values versus the actual SOC values. Each point on the plot represents a data instance from the test set. The x-axis represents the actual SOC values, while the y-axis represents the predicted SOC values obtained from the model. The points are color-coded in blue and have a transparency level of 0.3 to enhance visibility.

The black line in the plot represents the ideal scenario where the predicted and actual SOC values perfectly align. The scatter plot allows for a visual comparison between the predicted and actual values, enabling us to assess the performance of the model. If the points closely follow the black line, it indicates that the model is accurately capturing the patterns and variations in the data. Deviations from the black line may indicate the presence of biases or inconsistencies in the model's predictions.

The plot provides a visual representation of how well the model is performing in predicting the SOC values and allows for a better understanding of the model's accuracy and reliability. It helps in identifying any potential discrepancies or trends in the predictions, thereby aiding in the evaluation and interpretation of the model's performance.

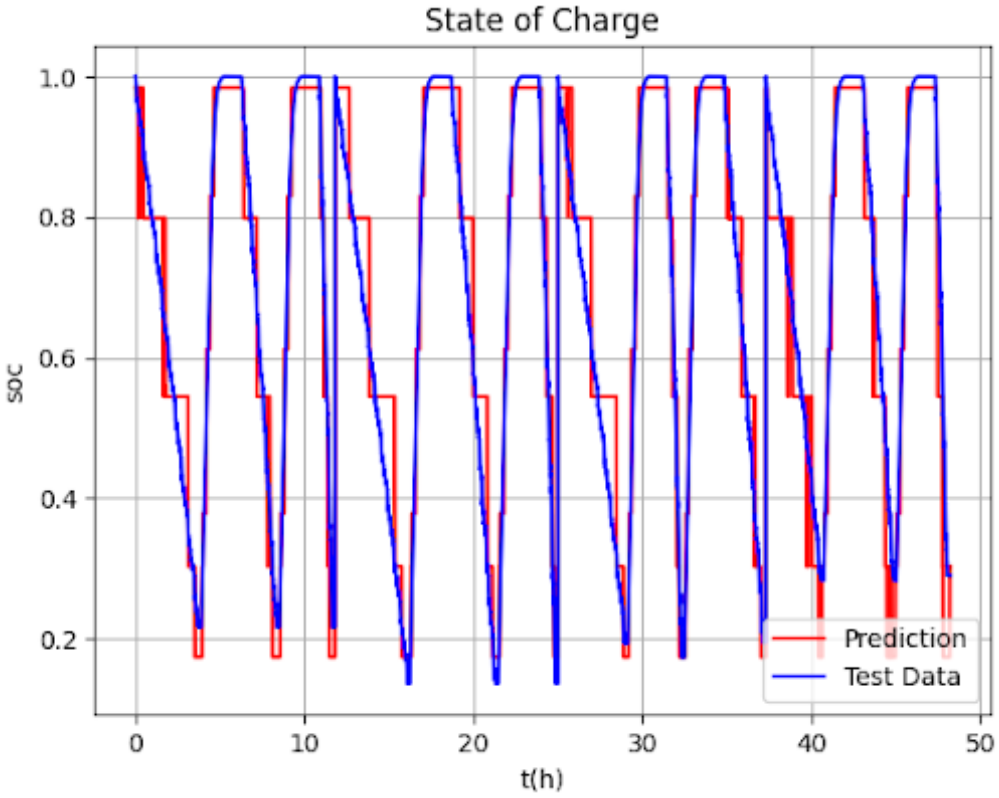

Figure 3.14: Comparison between soc of model 1 and test model 1

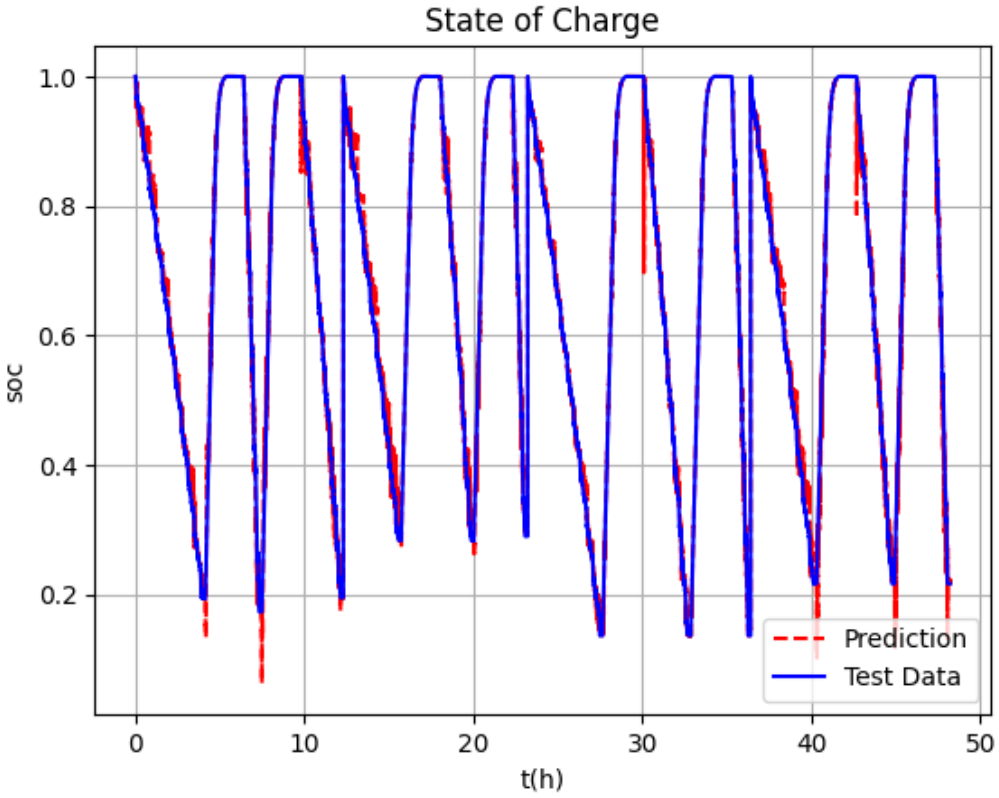

Figure 3.15: Comparison between soc of model 2 and test model 2

3.9 Conclusion

we explored the crucial steps of data preprocessing, model training, and the obtained results. We discussed the methodology for preparing the data, including conversion, handling missing values, outlier detection, and feature scaling. We then delved into the training process of the Random Forest Regression model, covering data splitting and the actual model training. We also discussed the deployment of the trained model through a user-friendly interface. Finally, we presented and discussed the results, considering dataset characteristics, feature selection, hyperparameter tuning, and model interpretability.

General Conclusion

In conclusion, this work has been a resounding success in developing a smart method for battery state prediction using machine learning techniques, specifically Random Forest Regression. Through meticulous data analysis, preprocessing, and model training, we have achieved remarkable results in accurately estimating the state of charge (SOC) of batteries. Our models have demonstrated high accuracy and reliability, as evidenced by the low mean squared error (MSE) and high R-squared values obtained during validation and testing.

The successful implementation of Random Forest Regression models in this study showcases their effectiveness in capturing the complex relationships and patterns within battery data, leading to precise SOC estimation. This accomplishment has significant implications for various applications reliant on battery management systems, such as electric vehicles and renewable energy systems. With improved SOC estimation, we can enhance battery utilization, optimize system performance, and ensure reliable operation.

However, it is important to acknowledge the limitations and challenges encountered during this research. While our models have delivered exceptional results, there is always room for improvement. Further investigation into feature selection, hyperparameter tuning, and model interpretability can potentially enhance the performance of the models and provide deeper insights into the underlying battery behavior.

Additionally, the availability of diverse and larger datasets can contribute to the generalizability and robustness of the models. Further research could involve incorporating real-time data acquisition and continuous model training to adapt to dynamic battery conditions.

Overall, the success of this work paves the way for future advancements in battery state prediction and underscores the significance of machine learning in addressing complex problems in battery management systems. The insights gained from this study contribute to the ongoing efforts in improving battery performance, efficiency, and overall sustainability.

Bibliography

[1] Chang, W. Y. (2013). The state of charge estimating methods for battery: A review. International Scholarly Research Notices, 2013.

[2] Xiong, R., Cao, J., Yu, Q., He, H., Sun, F. (2017). Critical review on the battery state of charge estimation methods for electric vehicles. Ieee Access, 6, 1832-1843.

[3] Habib, A.K.M.A.; Hasan, M.K.; Issa, G.F.; Singh, D.; Islam, S.; Ghazal, T.M. Lithium-Ion Battery Management System for Electric Vehicles: Constraints, Challenges, and Recommendations. Batteries 2023, 9, 152. https://doi.org/10.3390/batteries9030152

[4] Saxena, S., Sanchez, G., Pecht, M. (2017). Batteries in portable electronic devices: A user's perspective. IEEE Industrial Electronics Magazine, 11(2), 35-44.

[5] Arabul, F. K., Arabul, A. Y., Kumru, C. F., Boynuegri, A. R. (2017). Providing energy management of a fuel cell–battery–wind turbine–solar panel hybrid off grid smart home system. International journal of hydrogen energy, 42(43), 26906-26913.

[6] Marsh, R. A., Vukson, S., Surampudi, S., Ratnakumar, B. V., Smart, M. C., Manzo, M., Dalton, P. J. (2001). Li ion batteries for aerospace applications. Journal of power sources, 97, 25-27.

[7] Ben Amar, A., Kouki, A. B., Cao, H. (2015). Power approaches for implantable medical devices. sensors, 15(11), 28889-28914.

[8] Xing, Y., Ma, E. W., Tsui, K. L., Pecht, M. (2011). Battery management systems in electric and hybrid vehicles. Energies, 4(11), 1840-1857.

[9] Plett, G. L. (2015). Battery management systems, Volume II: Battery modeling (Vol. 2). Artech House.

[10] Pop, V., Bergveld, H. J., Danilov, D., Regtien, P. P., and Notten, P. H. (2008). Battery management systems: Accurate state-of-charge indication for battery-powered applications (Vol. 9). Springer Science and Business Media.

[11] https://www.linkedin.com/pulse/true-cost-lithium-ion-battery-systems-manaf-abdul

[12] Murnane, M., Ghazel, A. (2017).A closer look at state of charge (SOC) and state of health (SOH) estimation techniques for batteries . Analog devices, 2, 426-436.

[13] Zine, B., Bia, H., Benmouna, A., Becherif, M., and Iqbal, M. (2022). Experimentally validated coulomb counting method for battery state-of-charge estimation under variable current profiles. Energies, 15(21), 8172.

[14] Sundaresan, S., Devabattini, B. C., Kumar, P., Pattipati, K. R., Balasingam, B. (2022).Tabular open circuit voltage modelling of li-ion batteries for robust soc estimation . Energies, 15(23), 9142.

[15] L.Breiman,"Random Forest," Machine Learning,2001,vol.45,pp.5-32.

[16] M. Segal,"Machine Learning Benchmarks and Random Forest Regression," Center for Bioinformatics and Molecular Biostatistics, UC San Francisco,2004.

[17] Yu, Z., Huai, R., Xiao, L. (2015). State-of-charge estimation for lithium-ion batteries using a kalman filter based on local linearization. Energies, 8(8), 7854-7873.

[18] Li, C., Chen, Z., Cui, J., Wang, Y., and Zou, F. (2014, August). The lithium-ion battery state-of-charge estimation using random forest regression. In 2014 Prognostics and System Health Management Conference (PHM-2014 Hunan) (pp. 336-339). IEEE.

[19] Polamuri, S. R., Srinivas, K., and Mohan, A. K. (2019). Stock market prices prediction using random forest and extra tree regression. Int. J. Recent Technol. Eng, 8(1), 1224-1228.

[20] Khanvilkar, G., Vora, D. (2018). Sentiment analysis for product recommendation using random forest. International Journal of Engineering Technology, 7(3), 87-89.

[21] Wang, Z., Lai, C., Chen, X., Yang, B., Zhao, S., and Bai, X. (2015). Flood hazard risk assessment model based on random forest. Journal of Hydrology, 527, 1130-1141.

[22] Moussalli, Z., Sedra, M. B., Laachir, A. A. (2018, December). State of Charge estimation algorithms in Lithium-ion battery-powered Electric Vehicles. In 2018 International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS) (pp. 1-6). IEEE.

[23] https://fr.depositphotos.com/vector-images/pile.html

[24] Zhang, M., Fan, X. (2020).Review on the state of charge estimation methods for electric vehicle battery . World Electric Vehicle Journal, 11(1), 23.

[25] https://medium.com/@avinashkella/understanding-fuzzy-c-means-clustering-with-python-implementation-a-beginners-guide-3dbdf180393b

[26] https://why-change.com/2021/11/13/how-to-create-decision-trees-for-business-rules-analysis/

[27] https://www.ibm.com/fr-fr/topics/random-forest

[28] Lipu, M. S. H., Hannan, M. A., Hussain, A., Saad, M. H., Ayob, A., Blaabjerg, F. (2018). State of charge estimation for lithium-ion battery using recurrent NARX neural network model based lighting search algorithm. IEEE access, 6, 28150-28161.

[29] Xia, B., Cui, D., Sun, Z., Lao, Z., Zhang, R., Wang, W., Wang, M. (2018). State of charge estimation of lithium-ion batteries using optimized Levenberg-Marquardt wavelet neural network. Energy, 153, 694-705.

[30] Haenlein, M. and A. Kaplan, A brief history of artificial intelligence: On the past, present, and future of artificial intelligence. California management review, 2019. 61(4): p. 5-14.

[31] CHOUINI, M.E.M., Détection d'objets basé Faster R-CNN. 2020.

[32] LeCun, Y., Y. Bengio, and G. Hinton, Deep learning. nature, 2015. 521(7553): p. 436-444.

[33] Bhavsar, P., et al., Chapter 12 - Machine Learning in Transportation Data Analytics, in Data Analytics for Intelligent Transportation Systems, M. Chowdhury, A. Apon, and K. Dey, Editors. 2017, Elsevier. p. 283-307.

[34] Tourqui, I., Comparaison de méthodes de classification appliquées à la détection d'objets. 2016

[35] Géron, A. (2022). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. " O'Reilly Media, Inc.".

[36] https://machinelearningmastery.com/types-of-classification-in-machine-learning/

[37] https://why-change.com/2021/11/13/how-to-create-decision-trees-for-business-rules-analysis/

[38] Bruce, P., Bruce, A., and Gedeck, P. (2020). Practical statistics for data scientists: 50+ essential concepts using R and Python. O'Reilly Media.

[39] Delua, J. (2021). Supervised vs. unsupervised learning: What's the difference. IBM Analytics, 12.

[40] https://towardsdatascience.com/unsupervised-learning-and-data-clustering-eeecb78b422a

[41] Breiman, Leo. "Random forests." Machine learning 45 (2001): 5-32.

[42] https://python.plainenglish.io/do-standardization-and-normalization-transform-the-data-into-normal-distribution-cb5857ab9c63?gi=fe10

[43] https://www.ibm.com/topics/random-forest

[44] Fisher, Ronald Aylmer. "012: A Mathematical Examination of the Methods of Determining the Accuracy of an Observation by the Mean Error, and by the Mean Square Error." (1920).

[45] Chicco, Davide, Matthijs J. Warrens, and Giuseppe Jurman. "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation." PeerJ Computer Science 7 (2021): e623.

[46] https://www.analyticsvidhya.com/blog/2021/05/what-why-and-how-of-spectral-clustering/

[47] Kollmeyer, Phillip, et al. "Lg 18650HG2 li-ion battery data and example deep neural network xEV SOC estimator script." Mendeley Data 3 (2020): 2020.

[48] https://data.mendeley.com/datasets/cp3473x7xv/2.

[49] https://datascientest.com/en/python-the-most-popular-programming-language

[50] https://blog.boot.dev/python/matlab-vs-python/