sid.inpe.br/mtc-m21c/2018/04.27.23.39-TDI

MACHINE LEARNING SYSTEMS APPLIED IN SATELLITE LITHIUM-ION BATTERY SET IMPEDANCE ESTIMATION

Thiago Henrique Rizzi Donato

Master's Dissertation of the Graduate Course in Applied Computing, guided by Dr. Marcos Gonçalves Quiles, approved in April 02, 2018.

INPE
São José dos Campos
2018

PUBLISHED BY:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GBDIR)
Serviço de Informação e Documentação (SESID)
Caixa Postal 515 - CEP 12.245-970
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/6921
E-mail: pubtc@inpe.br

sid.inpe.br/mtc-m21c/2018/04.27.23.39-TDI

MACHINE LEARNING SYSTEMS APPLIED IN SATELLITE LITHIUM-ION BATTERY SET IMPEDANCE ESTIMATION

Thiago Henrique Rizzi Donato

Master's Dissertation of the Graduate Course in Applied Computing, guided by Dr. Marcos Gonçalves Quiles, approved in April 02, 2018.

URL of the original document:
<http://urlib.net/8JMKD3MGP3W34R/3R2BN4B>

INPE
São José dos Campos
2018

Cataloging in Publication Data

Donato, Thiago Henrique Rizzi.

D715m Machine learning systems applied in satellite lithium-ion battery set impedance estimation / Thiago Henrique Rizzi Donato. – São José dos Campos : INPE, 2018.

xxii + 62 p. ; (sid.inpe.br/mtc-m21c/2018/04.27.23.39-TDI)

Dissertation (Master in Applied Computing) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2018.

Guiding : Dr. Marcos Gonçalves Quiles.

1. Lithium-ion battery. 2. State of charge. 3. Gradient tree boosting. 4. Multi layer perceptron. I.Title.

CDU 621.352:004.032.26

Aluno (a): *Thiago Henrique Rizzi Donato*

Título: "MACHINE LEARNING SYSTEMS APPLIED IN SATELLITE LITHIUM-ION BATTERY SET IMPEDANCE ESTIMATION"

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de ***Mestre*** em ***Computação Aplicada***

Dr.	Rafael Duarte Coelho dos Santos	[illegible]
		Presidente / INPE / SJCampos - SP
		( ) Participação por Video - Conferência
Dr.	Marcos Gonçalves Quiles	[illegible]
		Orientador(a) / INPE / São José dos Campos - SP
		( ) Participação por Video - Conferência
Dr.	Reinaldo Roberto Rosa	[illegible]
		Membro da Banca / INPE / SJCampos - SP
		( ) Participação por Video - Conferência
Dr.	Elcio Hideiti Shiguemori	[illegible]
		Convidado(a) / IEAv / São José dos Campos - SP
		( ) Participação por Video - Conferência
Dr.	Wlamir Olivares Loesch Vianna	[illegible]
		Convidado(a) / EMBRAER / São José dos Campos - SP
		( ) Participação por Video - Conferência
Dr.	Márcio Porto Basgalupp	[illegible]
		Convidado(a) / UNIFESP / SJCampos - SP
		( ) Participação por Video - Conferência

Este trabalho foi aprovado por:

[✓] maioria simples

( ) unanimidade

São José dos Campos, 02 de abril de 2018

*To my family, work colleagues and fiancee, for
the infinite patience and unrelenting support*

ABSTRACT

In this work, the internal impedance of the lithium-ion battery pack, an essential measure of the degradation level of the batteries, is estimated employing ensembles of machine learning models. In this study, we take the supervised learning techniques Multi-Layer Perceptron bagging neural network and gradient tree boosting into account. Characteristics of the electric power system, in which the battery pack is inserted, are extracted and used in the modeling and training phases. During this process, the architecture of the ensembles and the configuration of their base learners are tuned through validation iterations. Finally, with the application of statistical testing and similarity analysis techniques, the best ensembles of models are examined and compared to other methods found in the literature. Results indicate that our approach is a suitable manner to estimate the internal impedance of batteries.

Keywords: Lithium-ion battery. State of charge. Gradient Tree Boosting. Multi Layer Perceptron.

ESTIMATIVA DA IMPEDÂNCIA DE CONJUNTOS DE BATERIAS DE LÍTIO-ÍON POR MEIO DE APRENDIZADO DE MÁQUINA

RESUMO

Neste trabalho, a impedância interna de um conjunto de baterias lítio-íon (uma importante medida do nível de degradação) é estimada por meio de conjuntos de modelos de aprendizado supervisionado tais como: rede neural tipo MLP (Multi-Layer Perceptron) e 'Gradient Tree Boosting'. Para isto, características do sistema de alimentação elétrica, em que o conjunto de baterias está inserido, são extraídas e utilizadas na construção de conjuntos de modelos supervisionados (MLP e xgBoost). Ao longo deste processo, a arquitetura de tais conjuntos de modelos e suas respectivas configurações são ajustados por meio de validações. Finalmente, com a aplicação de técnicas de teste e verificação estatística, as acurácias dos modelos são calculadas e testes comparativos são conduzidos. Os resultados obtidos mostram que a abordagem proposta é adequada para o problema de estimativa da impendância de baterias.

Palavras-chave: Bateria lítio-íon. Estado de carga. Gradient Tree Boosting. Multi Layer Perceptron.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ac	–	Alternate Current
AdNN	–	Adaptive Neural Network
AdNN	–	Adaptive Neural Network
AR	–	Auto Regressive model
AUC	–	Area Under ROC Curve
BHM	–	Battery Health Monitoring
CART	–	Classification And Regression Tree
CC	–	Constant Current
CNN	–	Condensed Nearest Neighbor
CNSA	–	China National Space Administration
CRISP-DM	–	Cross Industry Standard Process for Data Mining
CV	–	Constant Voltage
C-D	–	Charge and discharge cycle
EIS	–	Electrochemical Impedance
EKF	–	Extended Kalman Filter
EMF	–	Electromotive Force
EPS	–	Electrical Power Supply
ESA	–	European Space Agency
FP	–	False Positive
FPR	–	False Positive Rate
FN	–	False Negative
FNN	–	Fuzzy Neural Network
ML	–	Machine Learning
MLP	–	Multi Layer Perceptron
NASA	–	National Aeronautics and Space Administration
ND-AR	–	Nonlinear Degradation AutoRegression
OCV	–	Open Circuit Voltage
PHM	–	Prognostic and Health Management
PU	–	Per Unit
RBF	–	Radial Basis Function
RBFNN	–	Radial Basis Function Neural Network
RLM	–	Recursive Levenberg-Marquardt optimization method
RMSE	–	Root Mean Square Error
RRT	–	Remaining Run Time
SOC	–	State of Charge
SOH	–	State of Health
SVM	–	Support Vector Machine
TIEDVD	–	Time Interval of Equal Discharging Voltage Difference
VOC	–	Terminal Voltage of Battery
xgBoost	–	Gradient Tree Boosting

LIST OF SYMBOLS

α	–	learning rate
C_n	–	capacity of the battery at time t
η	–	current efficiency
I	–	current - assumed to be positive when charging
n	–	number of instances
ϕ'	–	derivative of the activation function
u	–	induced local field
y_t	–	observed instance at time t
$\hat{y}_t$	–	predicted instance at time t

CONTENTS

Page

1 INTRODUCTION

1.1 Satellite battery set actual challenges

Around 20–30 percent of a satellite total mass and a significant portion of its volume is due to the electrical power supply (EPS). Therefore, the EPS system is important to the success of a satellite mission. HILL (2011) enumerated the advantages and disadvantages of each technology and provided some parameters that allow the selection of the appropriate EPS system for a given application.

For outer space applications, Li-ion batteries improve the load efficiency of satellites since they are lighter (expected to have less than one half of the mass of nickel-hydrogen batteries for the same stored energy (DUDLEY, 1998)). As a result, Li-ion batteries have been used in satellites of National Aeronautics and Space Administration (NASA), China National Space Administration (CNSA) and Europe Space Agency (ESA) (CHIN; KEITH, 2018).

Failures of Li-ion battery can cause catastrophic consequences (overheating and even explosions). The study of techniques that aim to detect a failure causing damage to battery systems in early stages is denominated Prognostic and Health Management (PHM). The early detection of failures enables the minimization of the damage risk and the optimization of the Li-ion battery maintenance schedules (ZHANG; LEE, 2011).

The battery PHM methods include measurements of the battery remaining useful life through direct approaches, which demand high sensing, and machine learning approaches, which require low sensing (REZVANIZANIANI et al., 2014). The estimation of the battery remaining useful life is given by its state of charge (SOC) (SAHA et al., 2009).

The estimation of battery SOC can prevent major accidents. One example of such accidents occurred during the NASA's Mars Global Surveyor program. During that mission, after the adjustment of the satellite's solar panels position, the spacecraft was reoriented to an angle that exposed one of two batteries to direct sunlight. The overheating caused by the battery positioning accelerated the degradation process and led to the loss of both batteries (WEBSTER, 2007).

1.2 Li-ion battery state of charge (SOC) estimation

The *SOC* is crucial to obtain the battery remaining useful life. According to (WATRIN et al., 2012), *SOC* estimation methods are divided into the following categories (see Chapter 3).

- Direct measurement: applies physical battery properties, such as the voltage and impedance of the battery (see Section 3.1), and requires heavy sensing.
- Machine learning models: applies indirect battery parameters and can automatically adjust the *SOC* for different discharging conditions (see Section 3.2).
- Hybrid methods: benefit from the advantages of both *SOC* estimation approach and allow a better estimation of battery *SOC*. According to (WATRIN et al., 2012), hybrid methods generally produce more accurate estimation of *SOC*, compared to individual methods (see Section 3.3).

Although there is a high variety of state of charge (*SOC*) estimation methods, we still have room for improvements. In this study, we developed two ensembles of machine learning models (neural network - Section 2.3.1 and decision tree - Section 2.3.2.1) which aim to estimate the Li-ion battery internal impedance which is directly correlated to its *SOC* (according to Section 3.4).

To perform the comparison between the two machine learning ensembles of models, the Li-ion battery testing database provided by NASA Ames Research Center was taken into account. It comprises sensor monitoring data of Li-ion batteries running through 3 different operational profiles (charge, discharge and impedance measurement) at the ambient temperature of 24°C (SAHA; GOEBEL, 2007). The laboratory setup is described in Section 4.1.1.

To characterize the battery condition during a certain period, features introduced in Section 3.5 were calculated. Then, ensembles of machine learning models were trained (see Section 4.4), and the best configurations of the hyperparameters were obtained through validation iterations. Finally, a performance metric was used to compare them (see Section 5.3) with each other and with a benchmark regression model applied to the same battery data set (see Section 5.3.3).

The benchmark approach used in this comparison is according to the following paper: A Comparative Analysis of Techniques for Electric Vehicle Battery Prognostics and Health Management (REZVANIZANIANI et al., 2011). This paper presents a comparative study of emerging prognostics and health management techniques that can give an accurate quantification of the state of health of Li-ion battery cells and predict their remaining useful life. Adaptive Neural Network ($AdNN$) (see Section 3.2.2.3) is used for battery internal impedance estimation and remaining useful life prediction. Its prediction performance is benchmarked using Li-ion battery dataset (SAHA; GOEBEL, 2007).

The present study applies one-sample Student's t-test (see Section 2.5) in the evaluation of how significantly different from each other are the performance metrics of the obtained ensembles of models. In addition, this study also applies one-sample Student's t-test (see Section 2.5) in the evaluation of how significantly different are the performance metrics of these ensembles of models when compared with the benchmark regression model.

This document is organized as it follows. Chapter 2 provides background about the application of machine learning models in regression. Chapter 3 provides background about the details of benchmark Li-ion battery state of charge estimation techniques. Chapter 4 describes the laboratory setup, data preparation, and data modeling. Chapter 5 depicts the obtained results from the application of ensembles of machine learning supervised models in the Li-ion battery impedance estimation. This chapter also presents the performance metrics and the statistic difference between the proposed ensembles and the benchmark model. Finally, concluding remarks are drawn in Chapter 6.

2 MACHINE LEARNING IN REGRESSION

Machine learning models are suitable computational models for detecting changes in system states (LIU; WANG, 2009). Following this approach, a few information regarding the analyzed system is necessary to estimate a future condition of the system. This characteristic is essential to EPS systems which are sufficiently complex prohibiting the development of an accurate physical model. However, this approach implies in wider confidence intervals than other approaches and requires a substantial amount of data for training.

After the data preparation, a machine learning algorithm shall be selected to build the model. The selection of a proper algorithm for a specific application is a challenging factor in applying data-driven prognostics methods (ROBERT, 2014).

In this study, the goal is to estimate Li-ion battery internal impedance which is related to Li-ion battery SOC. Considering that the target variable is numerical, the following Sections describe relevant concepts and definitions regarding the machine learning regression methods which are used by the regression ensembles suggested by this study and detailed in Chapter 4.

2.1 Regression fundamentals

Regression models allow the estimation of a dependent variable (target) from the analysis of a few independent variables which are related to the problem. To obtain a function linking the target with the independent variables, in the most of the cases, only a sample of the population is available. Therefore, the analyzed sample shall be representative for the prediction of the target. This sample is denominated as training set and, through the function which relates the independent variables with the target, further target values can be inferred.

In order to build an adequate regression model, the independent variables shall be measured accurately, since measurement errors can lead to disturbs in the target estimation, and the ones which are a linear combination of others shall be excluded from the training set (ARMSTRONG, 2012).

The function which represents the regression model relates the independent variables (X) weighted by the vector α with the dependent variable (Y) as follows:

$$\mathbf{Y} \approx f(\mathbf{X}, \boldsymbol{\alpha}) \tag{2.1}$$

According to the equation, the vector of unknown parameters α shall be defined to obtain the regression model which allows the definition of the target variable. To define α, a number N of training points must be available, and the target variable shall be known for these training points.

If the number N of training points is equal to the dimension k of vector α, the function f is linear, and the independent variables are linearly independent, the target variable can be exactly obtained. In this case, α is computed by solving a set of N equations.

If the number N of training points is lower than the dimension k of vector α, the regression analysis cannot be applied. However, case the number N of training points is higher than the dimension k of vector α, the system is overdetermined and the vector α can be calculated to obtain the function f which best fits the training set. This function is the regression model, and its quality can be achieved through the distance between the predicted values, and the real values of the target variable (in Section 2.4.3 the performance metrics are detailed). The techniques proposed in this study use simple machine learning models which, when combined, achieve a good result when applied to the regression problem detailed herein. The combination of simple machine learning models (denominated as base learners) is called ensemble and the main methods used to combine base learners are described in Section 2.2.

2.2 Ensemble of machine learning models

A machine learning model is suitable for a specific problem. In the development of the model, a hypothesis or premise shall be considered. From the adopted hypothesis, the model is elaborated, and its performance can be tested. However, the obtained result is related to the premise.

Empirically, the combination of various models (each one considering a distinct premise), tends to yield better results when there is significant diversity among the models (KUNCHEVA; WHITAKER, 2003).

Due to the high computation needed in the development and testing of the various models which form the ensemble, these models generally have a simple architecture since the accurate final performance is achieved by an adequate combination of them (denominated as base learners). Therefore, these base learners are usually denominated as "weak learners".

In the following sections, two ensembles techniques considered in this study are details.

2.2.1 Bagging

The bagging ensemble strategy consists of combining the results of the "weak learners" through their aggregation with equal weight. Therefore, to reduce the variance, each model of the ensemble uses a random subset of instances from the training set. This approach minimizes the influence of outliers since in most of the models these instances are not contained in the training selected subset (BREIMAN, 2003).

According to bagging ensemble strategy, a training set D of size n is divided into m new training sets D_i, each of size n, by sampling from D uniformly and with replacement. Sampling the m new training sets with replacement implies that some observations may be repeated in each D_i (ASLAM et al., 2007).

In order to obtain the resulting ensemble of models, the m models are fitted using the m bootstrap samples. Finally, all the models are applied to the scoring set, and the labels are combined by averaging the output (for regression) or voting (for classification).

This study applies the bagging ensemble method in the aggregation of the results obtained with the application of a set of "weak learners" developed by the MLP neural network technique which is described in Section 2.3.1.

2.2.2 Boosting

The boosting ensemble strategy incrementally creates "weak learners" which aim to reduce the classification error of the previous models by emphasizing the training instances that were previously misclassified. Generally, decision tree technique is applied in the development of the "weak learners" in boosting ensembles due to its simplicity.

2.2.2.1 AdaBoost

The most popular boosting ensemble technique is the *AdaBoost* which, at each iteration, builds a "weak learner" weighted by the coefficient η_t such that the sum training error E_t of the resulting t-stage is minimized (KéGL, 2013).

$$E_t = E[F_{t-1}(x) + \eta_t h(x)] \tag{2.2}$$

Where:

- x is the collection of training instances
- $F_{t-1}(x)$ is the boosted classifier built at the previous stage of training
- $E(F)$ is the error function
- η_t is the learning rate in time t which weights the weak learner to be added to the final classifier
- $f_t(x) = \eta_t h(x)$ is the weak learner that is being considered to add to the final classifier

2.2.2.2 Gradient boosting

The gradient boosting technique creates a sequence of "weak learners" (assembled in a model F) which intends to minimize the mean squared error (FRIEDMAN, 2000). Therefore, after every iteration of the gradient boosting process, it builds a new model from the previous one which includes an estimator $h(x)$ to enhance the resulting model:

$$F_{m+1}(x) = F_m(x) + h(x) \tag{2.3}$$

The estimator $h(x)$ is calculated assuming that a perfect $h(x)$ would fit exactly the target variable:

$$F_{m+1}(x) = F_m(x) + h(x) = y \tag{2.4}$$

Considering the perfect estimator as $h(x) = y - F_m(x)$, minimizing the residual $y - F_m(x)$ implies in approaching the best estimator. Recursively, each F_{m+1} model corrects the previous one F_m.

The problem of minimizing the residual $y - F(x)$ for a given model consists of the negative gradients (concerning $F(x)$) of the squared error loss function $\frac{1}{2}(y - F(x))^2$. Therefore, gradient boosting is a gradient descent algorithm (SCHAPIRE; FREUND, 2012). This study applies the gradient boosting ensemble method in the aggregation of the results obtained with the application of a set of "weak learners" developed by the decision tree technique which is described in Section 2.3.2.

2.3 Machine learning techniques

This study applies two ensembles of models in order to estimate the battery internal impedance: a MLP neural network bagging and a gradient tree boosting. Each of these ensembles is generated by "weak learners" developed by MLP neural network (see Section 2.3.1) and Gradient tree boosting (see Section 2.3.2) techniques, respectively.

2.3.1 Multi Layer Perceptron neural network bagging

The MLP neural network bagging consists of an ensemble of "weak learners" which are artificial neural networks structured according to the MLP structure. Each one of these learners selects a subset of the training set randomly, and the final result is obtained through the simple average of the base learners outputs.

There are several types of artificial neural networks applied in regression problems, as exemplified in Section 3.2.2. Among them, the Multi-Layer Perceptron neural network is the most popular technique.

The MLP neural network is a directed graph divided into layers of nodes. The neurons of a layer are fully connected to the neurons of the next layer, except for the input nodes which are the input data itself. The intermediary layers and the last layer contain neurons that apply a nonlinear activation function to the input sign and generate an output (KRUSE, 2013).

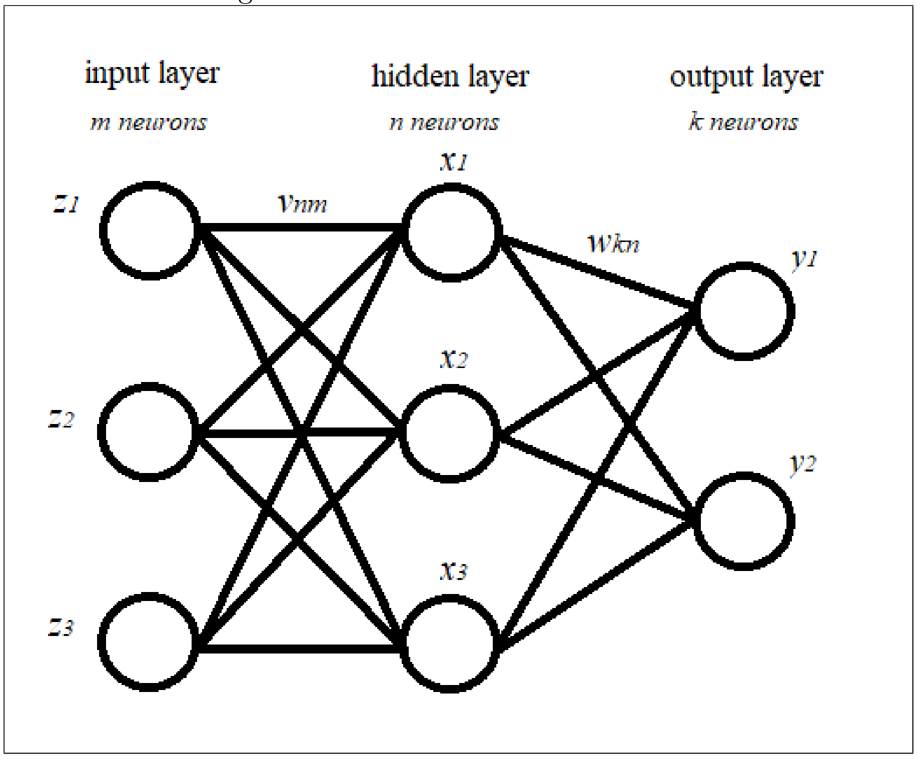
Figure 2.1 illustrates an example of a MLP neural network. The z_m neurons (input layer) correspond to the predictors values of the training matrix which are weighted by the weight vector v_{nm} and linked to the x_n neurons of the hidden layer. These neurons apply the activation function to their inputs whose results are weighted by the weight vector w_{kn} and linked to the output layer y_k (output layer).

2.3.1.1 Multi Layer Perceptron activation function

Since each neuron of the MLP neural network applies a nonlinear activation function to the weighted inputs, the network can distinguish data that are not linearly separable (CYBENKO, 1989).

The two most common activation functions applied by neurons of an MLP neural network are both sigmoids and, using the output layer of the MLP according to Figure 2.1 as an example, the activation functions are described by:

Figure 2.1 - MLP network architecture



Source: (HAYKIN, 1999)

$$Y_k = \tanh(w_{kn}) \quad \text{and} \quad Y_k = (1 + e^{-w_{kn}})^{-1}, \tag{2.5}$$

Where:

- z_m: neuron in position m of the input layer
- x_n: neuron in position n of the hidden layer
- y_k: neuron in position k of the output layer
- Y_k: output of the neuron in position k of the output layer
- v_{nm}: weighted output of neuron z_m of the input layer which generates an input of neuron x_n of the hidden layer
- w_{kn}: weighted output of neuron x_n of the hidden layer which generates an input of neuron y_k of the output layer

The first one is denominated as the hyperbolic tangent function which ranges from -1 to 1, and the second one is denominated as the logistic function which is similar in shape but ranges from 0 to 1.

Here y_m is the output of the k_{th} neuron and w_{kn} is the weighted sum of the input synapses. In specific applications, other activation functions can be selected, like the rectifier and softplus functions.

In the present study, it is used the *tanh* activation function due to the following requirements:

- The neural network weights are obtained through the backpropagation technique which requires the activation function to be differentiable, smooth, monotonic and bounded
- The activation function shall not cause the undesirable behavior of flatting the error surface after some iterations of the learning process

The logistic function complies with the first requirement once corresponds to a monotonically increasing function. However, if the argument of the function is substantially negative, it generates output values too close to 0.0 which leads to a low learning rate for all subsequent weights and an ineffective continuation of the training process. On the other hand, the *tanh* function will, in the same situation, generate a value close to -1.0, and maintain learning rate (LECUN et al., 1998).

2.3.1.2 Multi Layer Perceptron learning algorithm

MLP network is initialized with random weights (preferably values close to zero). During the neural network training process, the learning algorithm updates the network weights until it achieves a behavior identified in the training set. During each iteration of this iterative process, an example or a subset of the training set is presented to the network which uses its independent variables values as input and its target values as output. To train the network by using these input and output data, it is applied a supervised learning function called backpropagation (RUMELHART et al., 1986).

During the learning process, the backpropagation method updates the network weights based on the amount of error in the output compared to the expected result (the label value). Therefore, after each iteration, the learning process minimizes the least mean squares.

Using the output layer of the MLP according to Figure 2.1, the error in output node y_k in the i_{th} data point (training example) by $e_k(i) = d_k(i) - y_k(i)$, where d is the target value, and y is the value produced by the network. The corrections to the weights of the nodes are made based on those corrections which minimize the error in the entire output, given by:

$$\mathcal{E}(i) = \frac{1}{2}\sum_k e_k^2(i). \tag{2.6}$$

Using gradient descent, we find our change in each weight to be

$$\Delta w_{k_n}(i) = -\frac{\partial \mathcal{E}(i)}{\partial u_k(i)} x_n(i) * \alpha \tag{2.7}$$

Where:

- x_n: output of the previous neuron
- α: learning rate (defines how fast the weights converge to a response)
- u_k: induced local field

The sum of the weighted inputs of neuron y_k is the induced local field u_k. For an output node the derivative of formula 2.7 can be simplified by:

$$-\frac{\partial \mathcal{E}(i)}{\partial u_k(i)} = e_k(i)\phi'(u_k(i)) \tag{2.8}$$

Where:

- ϕ': derivative of the activation function

For a node located in a hidden layer the relevant derivative is calculated as below:

$$-\frac{\partial \mathcal{E}(i)}{\partial u_n(i)} = \phi'(u_n(i))\sum_k -\frac{\partial \mathcal{E}(i)}{\partial u_k(i)} w_{k_n}(i). \tag{2.9}$$

According to the formula 2.9, the update of the n_{th} hidden nodes weights depends on the change in weights of the k_{th} nodes, which represent the output layer. Therefore, to update the hidden layer weights, the output layer weights are defined previously according to the derivative of the activation function. This algorithm is denominated backpropagation due to the direction of the weights updating which occurs from the output layer to the input layer (HAYKIN, 1999, Chapter 4).

2.3.1.3 Multi Layer Perceptron learning momentum

According to the backpropagation algorithm, the MLP network initiates with random weights at a specific point on the error function and updates the weights recursively until the achievement of a global minimum of the error function. However, according to Equation 2.10, the global minimum can be equal to a local minimum or the global minimum can be different from the local minimum. Therefore, a modification of the basic backpropagation learning algorithm is conducted to avoid the convergence to a local minimum (SHEEL et al., 2007).

$$\Delta we_{kn}(t+1) = (1-\alpha)\eta * \delta_n * w_{kn} + \alpha\,\Delta we_{kn}(t) \tag{2.10}$$

Where:

- $\Delta we_{kn}(t+1)$: change in weight of w_{kn} in time $t+1$
- $\Delta we_{kn}(t)$: change in weight of w_{kn} in time t
- we_{kn}: weight which multiplies the output of neuron x_n of the hidden layer which generates an input of neuron y_k of the output layer at time $t+1$
- η: learning rate
- α: the influence of the inertial term
- δ_n: error signal of neuron x_n
- x_n: neuron in position n of the hidden layer
- w_{kn}: weighted output of neuron x_n of the hidden layer which generates an input of neuron y_k of the output layer

2.3.2 Gradient tree boosting

Gradient tree boosting is an ensemble method which aims to create an optimal regression model combining various decision trees ("weak learners"). Each one of these learners is built from the previous one and aims to minimize the error of the prior learner through gradient boosting method (according to Section 2.3.2.2).

2.3.2.1 Decision tree

A decision tree consists of a set of rules each of them dividing the training set into subsets which are further divided into other subsets according to the remaining rules.

As an example, a simple decision tree is built to estimate the mean Li-ion battery internal impedance during a particular cycle of charge and discharge. The training set according to Table 2.1 details the values of features F1 and F2 (described in Section 3.5) of the battery during four initial charge and discharge cycles.

Table 2.1 - Battery simple training set

N Cycle	F1	F2	Impedance
1	27.0	25.7	0.061
2	25.6	25.7	0.059
3	25.6	26.2	0.059
4	25.2	25.9	0.059

To create a simple decision tree to estimate the mean Li-ion battery internal impedance at the fifth charge and discharge cycle, one of the features $F1$ or $F2$ is selected to be the top of the decision tree. To select this first attribute that generates a tree, a measurement of entropy is applied to verify what feature is able to divide the training set into the most homogeneous subsets regarding the target variable (BREIMAN et al., 1984). One of the most popular entropy indicators is the "Gini index". According to this entropy metric, the homogeneous degree of a training subset is the sum of the probability of its individuals of belonging to a specific class multiplied by the sum of the probabilities of belonging to the other classes, according to the Equation 2.11:

$$I_G(F_n) = \sum_{i=1}^{J} p_i \sum_{k\neq i}^{L} p_k = \sum_{i=1}^{J} p_i(1-p_i) = 1 - \sum_{i=1}^{J} {p_i}^2 \tag{2.11}$$

Where:

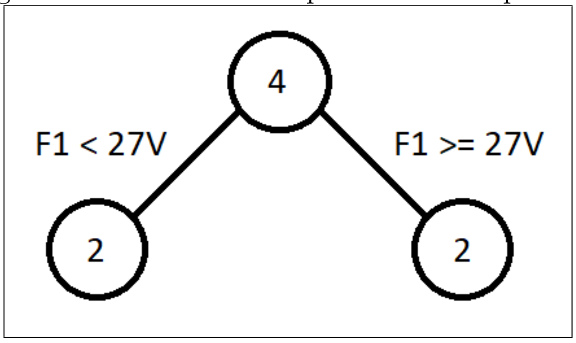
- F_n: feature n of the training set

- $I_G(F)$: Gini index of feature F_n
- p_i: probability of i training instance of belonging to a specific class
- p_k: probability of i training instance of belonging to a distinct class from the actual instance class
- $\sum_{k \neq i}^{L} p_k$: sum of the probabilities of i training instance of belonging to the other classes

Applying the training set according to Table 2.1, two simple decision trees of only two branches each (denominated decision stump) can be built to estimate the mean Li-ion battery internal impedance.

The first decision tree selects the feature $F1$ as the top feature of the decision tree and chooses the threshold equal to $27V$ to divide the training set into two subsets. Therefore, the obtained decision stump is according to Figure 2.2.

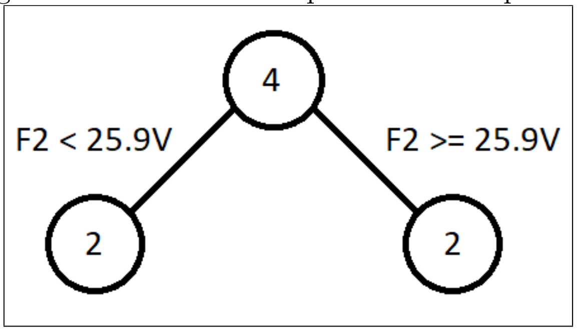
Figure 2.2 - Decision stump with F1 as top feature



This decision stump divides the training set which contains four individuals into two distinct subsets each one containing two individuals. All the individuals with the value of $F1$ feature above or equal to $27V$ have the impedance value of 0.061Ω, and all the individuals with the value of $F1$ feature below $27V$ have the impedance value of 0.059Ω. For this reason, the decision tree subsets are homogeneous, and the decision stump can estimate accurately the mean Li-ion battery internal impedance.

The second decision tree selects the feature $F2$ as the top feature and chooses the threshold equal to $25.9V$ to divide the training set into two subsets. Therefore, the obtained decision stump is according to Figure 2.3.

Figure 2.3 - Decision stump with F2 as top feature



This decision stump divides the training set which contains four individuals into two distinct subsets each one containing two individuals. All the individuals with the value of $F2$ feature above or equal to $25.9V$ have the impedance value of 0.059Ω. However, only one individual with the value of $F2$ feature below $25.9V$ has the impedance value of 0.061Ω. The other individual of the second group has the impedance value of 0.059Ω. For this reason, the decision tree subsets are not entirely homogeneous (the second subset is fifty percent homogeneous) and the decision stump cannot estimate accurately the mean Li-ion battery internal impedance. In other words, the decision stump created with the subset by feature $F2$ has a lower accuracy when compared with the decision stump created from the feature $F1$.

Applying an entropy measurement technique before selecting the top feature to begin splitting the training set, the feature which splits the individuals into the most homogeneous subsets can be selected and the optimized decision stump is obtained.

In this example, the decision tree contains one level (only one split is executed). However, the decision tree can grow by splitting the obtained subsets. At each decision tree level, the algorithm identifies the most relevant feature to be used in the split and divides the subset into two other subsets which are divided during further splits.

2.3.2.2 Gradient tree boosting ensemble

The gradient tree boosting algorithm intends to minimize the mean squared error through gradient boosting (see Section 2.3.2) applying decision trees as the "weak learners". During the boosting process, a decision tree aims to minimize the mean squared error from the previous one.

After computing $F_t, F_{t-1}, F_{t-2}...$, the decision trees split the input space into J_t disjoint regions $R_{1t}, \dots, R_{J_t t}$ and obtain a constant value in each region. The combination of the tree partitions outputs is obtained as follows:

$$H_t(x) = \sum_{j=1}^{J_t} b_{jt} I(x \in R_{jt}), \tag{2.12}$$

Where:

- $H_t(x)$ is the ensemble of decision tree models obtained through the application of instances x at iteration t
- b_{jt} is the value predicted in the region R_{jt}
- R_{jt} is the region which contains all the training features

The combination of the tree partitions depends on the learning rate η_t. Increasing the learning rate, the time until the convergence reduces. However, a high learning rate increases the risk of not achieving the convergence. After selecting the learning rate, the model is updated as follows (HASTIE; FRIEDMAN, 2009):

$$F_t(x) = F_{t-1}(x) + \eta_t h(x), \quad \eta_t = \arg\min_{\eta} \sum_{i=1}^{n} L(F_t(x), F_{t-1}(x_i) + \eta_t h(x_i)) \tag{2.13}$$

Where:

- x is the collection of n training instances
- $F_t(x)$ is the gradient tree boosting classifier built at the actual stage of training
- $F_{t-1}(x)$ is the gradient tree boosting classifier built at the previous stage of training
- η_t is the learning rate in time t which weights the weak learner to be added to the final classifier
- $h_t(x) = \eta_t h(x)$ is the weak learner that is being considered to add to the final classifier

2.3.2.3 Gradient tree boosting features importance

As detailed in Section 2.3.2.1, during the development of each decision tree, which forms the gradient tree boosting ensemble, at each decision tree level, the feature that splits the training data into the most homogeneous subsets is calculated. Therefore, the features used to split the training data at the decision top-three levels are considered the most relevant features since they can split the training set into the larger homogeneous subsets (BREIMAN et al., 1984). The importance of the features in a decision tree is considered during the simplification of the model. Specifically, one might remove the less important features with a low degradation.

2.4 Validation and testing

2.4.1 Validation

During the modeling stage, the hyperparameters of the ensembles of models are tuned within validation iterations. During each validation iteration, a specific ensemble of models built with a certain hyperparameters configuration is applied to a validation set (the subset of the historical dataset apart from the training set), and the performance metric is calculated (herein it is applied the $RMSE$ performance metric). Finally, after a few validation iterations, the ensembles of models which achieved the lower validation error (according to the applied performance metric) are considered the most accurate ensembles of models.

To increase the chances that the ensembles of models built with the training set will generalize to an independent dataset and continue to achieve an adequate result in a real problem with low risk of overfitting the training and validation sets, a statistical analysis denominated cross-validation can be performed.

The cross-validation technique consists of splitting the training set into a certain number of folders. Each folder is a partition of the training set and, considering one round cross-validation, one of the folders is segregated as a validation set while the remaining folders are used to train the ensemble of models. Thus, the obtained ensemble of models is validated with the computation of $RMSE$ performance metric applying the ensemble of models to the validation set (RON, 1995).

Considering a two round cross-validation, two validation iterations are performed and, during each iteration, a specific folder is used as validation set and the remaining ones as training set. During each iteration a $RMSE$ performance metric is obtained and the mean $RMSE$ performance metric is the result of the cross-validation. The number of rounds of the cross-validation process is limited to the number of folds (splits of the original training set).

2.4.2 Testing

After obtaining the best ensembles of models through the cross-validation iterations described in Section 4.4, a testing set completely apart from the training and validation sets is obtained from the historical database and used to verify the test $RMSE$ performance metric of the best ensembles of models. During the testing process, the ensembles of models are no longer tuned (LORENZO et al., 2015). The test $RMSE$ performance metric indicates how well the ensembles of models perform in the scoring data (the data to be predicted with the ensembles of models application). Thus, the obtained result can be compared with other classification approaches.

2.4.3 Performance metric

Among a variety of performance metrics, the Root Mean Square Error ($RMSE$) technique is broadly applied in the validation, and testing of regression models. The $RMSE$ performance metric is used herein in the validation stage when the ensembles of models are tuned, and the best hyperparameters configuration is obtained, and in the testing stage, when the best-obtained ensembles of regression models are compared with each other and with the benchmark approach.

Considering a numeric label attribute in the estimative of the battery set impedance, each observed value of the validation or testing set can be compared with the predicted one. This individual deviation is called a residual and the aggregation of all the residuals is obtained as follows (HYNDMAN; KOEHLER, 2006):

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{n}(\hat{y}_t - y_t)^2}{n}} \tag{2.14}$$

Where:

- $\hat{y}_t$: predicted instance
- y_t: observed instance
- n: number of instances

2.5 Statistical similarity

In this study, the obtained results are compared with each other and with a benchmark approach. Therefore, a measurement of statistical similarity is applied to evaluate if the performance metrics are significantly different from each other. Thus, it is used the Student's t-test which is most commonly applied when the compared subsets follow a normal distribution (FADEM, 2008).

This present study applies the one-sample Student's t-test, as described below:

$$Z = \frac{(\bar{X} - \mu)}{(\frac{\sigma}{\sqrt{n}})} (2.15)$$

Where:

- X is the sample mean from a sample $X1, X2, \ldots, Xn$, of size n
- σ is the population standard deviation of the data
- μ is the population mean

Once the Z value is calculated, in order to define if X is significantly different from σ, it shall be compared with a predefined threshold. Case the Z value is above the threshold, the statistic difference is validated and the sample mean can be considered distinct of the population mean.

3 LI-ION BATTERY SOC ESTIMATION

The estimation of the state of charge is critical in the design of charging/discharging operational cycles (cell-level balancing) and in the elaboration of a reliability program to prevent the occurrence of abnormalities such as overcharging, overheating and over-discharge. The accurate SOC estimation leads to a reliable definition of the remaining useful energy, the optimization of the battery operation and the extension of its operational life (ZHANG; LEE, 2011).

SOC is calculated according to Equation 3.1. The nominal capacity (Q_n) is the maximum energy stored by the battery in its initial operational cycles and it is given by the manufacturer.

$$SOC_t = \frac{Q_t}{Q_n}. \tag{3.1}$$

Therefore, once the nominal capacity is already defined, the methods described below intend to estimate the actual battery capacity after charge/discharge cycles.

According to (WATRIN et al., 2012), the various mathematical methods of estimation are classified in the following categories:

3.1 Direct measurements

Direct measurement methods use physical battery properties such as the terminal voltage and impedance in the estimation of SOC and usually require heavy sensing.

3.1.1 Open circuit voltage method

For lead-acid battery, there is a linear relationship between the SOC and its open circuit voltage (OCV) given by (COLEMAN et al., 2007):

$$OCV(t) = a_1 * SOC(t) + a_0. \tag{3.2}$$

Where:

- $OCV(t)$ is the open circuit voltage of the battery at t
- $SOC(t)$ is the state of charge of the battery at t

- a_0 is the battery terminal voltage when $SOC = 0$

- a_1 is obtained from the value of a_0

3.1.2 Terminal voltage method

For lead-acid battery, there is a approximate linear relationship between the SOC and its electromotive force (EMF). However, for Li-ion batteries, the terminal voltage of battery drops not linearly after the initial operation cycles and, therefore, it is not indicated to battery $SOC(t)$ estimation (SATO; KAWAMURA, 2002).

3.1.3 Impedance spectroscopy method

The impedance spectroscopy method is conducted under controlled experimental conditions and, for different charge and discharge currents, measures the battery internal impedance values over a wide range of frequencies (LI et al., 2010). Once there is a linear relationship between the battery SOC and its internal impedance, the state of charge under different operational conditions can be obtained. However, this method is intrusive and demands high sensing which is not always available in operational systems.

3.1.4 Coulomb counting method

The Coulomb counting method measures the discharging current of a battery and integrates the discharging current over time in order to estimate SOC by the following equation:

$$SOC(t) = SOC(t-1) + \frac{I(t)}{Q_n} * \Delta_t. \tag{3.3}$$

Where:

- $SOC(t)$ is the SOC value at time t

- $I(t)$ is the discharging current at time t

- Q_n is the battery nominal capacity

- Δ_t is the difference between t and $t-1$

3.2 Machine learning systems

The machine learning systems are self-designing ones that can be automatically adjusted in non-stationary systems. Batteries are affected by many chemical factors and have nonlinear SOC. Therefore, machine learning systems offer a good solution for SOC estimation.

In (SCHWABACHER, 2005), there are evaluated various machine learning algorithms applied in machine learning system and presents useful information to conclude pros and cons of each one. Examples of machine learning algorithms: multi layer perceptron neural network (MLP), gradient boosting ($xgBoost$), radial basis function (RBF), fuzzy logic methods, support vector machine (SVM), fuzzy neural network, and Kalman filter.

For batteries already in operation, the data collected during tests performed with similar equipment can be used as training set. In these cases, the efficacy of the machine learning system will be determined not only by the quantity of data but also by the quality of them. The data quality depends on the performed tests and consists of the presence or absence of noisy and high-dimensional data. To improve the data quality, there are techniques to extract representative features from multi-dimensional data and signals obscured by noise (MOSALLAM et al., 2013).

The following Sections describe machine learning techniques commonly applied in SOC estimation.

3.2.0.1 Auto regressive model

An autoregressive model consists of a time series analysis based on stochastic process theory which is applied in signal processing, intelligent information analysis, and PHM (PANDIT; WU, 1983).

An auto regressive model of order p is denoted by $AR(p)$ and defined as $X_t = c+\sum_{i=1}^{p} \varphi_i X_{t-i}+\varepsilon_t$, where $\varphi_1, \ldots, \varphi_p$ are the parameters of the model, c is a constant, and ε_t is white noise. In a matrix format, this is equivalent to $X_t = c+\sum_{i=1}^{p} \varphi_i B^i X_t+ \varepsilon_t$ so that, moving the summation term to the left side, we have $\phi(B)X_t = c + \varepsilon_t$ (PANDIT; WU, 1983).

In order to maintain the model wide-sense stationary, some parameter constraints are needed. For example, processes in the $AR(1)$ model with $|\varphi_1| \geq 1$ are not stationary. To assure $AR(p)$ model to be wide-sense stationary, the roots of the polynomial $z^p - \sum_{i=1}^{p} \varphi_i z^{p-i}$ must be inside of the unit circle. In other words, each (complex) root z_i must satisfy $|z_i| < 1$ (HASTIE et al., 2009).

According to (DATONG et al., 2012), it is applied AR model (Auto Regressive) and $ND - AR$ model (Nonlinear Degradation Auto Regression) for Remaining Useful Life (RUL) estimation of lithium-ion batteries based on the estimation of its internal impedance.

3.2.1 Gradient tree boosting

Gradient tree boosting popularity is due to the properties that all tree-based algorithms have: they can handle data of mixed type (continues, categorical, etc.) as well as data with missing values, they are immune to the presence of outliers as well as to the presence of irrelevant inputs, and they scale nicely. On the other hand, their predictive power is usually inferior compared to another state of the art ML algorithms, such as neural networks and $SVMs$ (HASTIE et al., 2009).

However, tree-based methods can produce highly accurate predictions if grouped in the form of an ensemble. A gradient tree boosting based SOC indicator is used to estimate the RUL of the battery during a flight mission which is based on the measurement of the terminal voltage (MANSOURI et al., 2017).

Section 2.3.2 details the gradient boosting ensemble of "weak learners" based on decision trees.

3.2.2 Artificial neural network

An artificial neural network (ANN) is a network of neurons or nodes linked one to another. These links allow the transmission of information between them. In this process, the input of a neuron is biased by the weight assign to this link. After, the output of the neurons is computed by a mathematical function named activation function. This output is propagated to the following neurons in the network (HAYKIN, 1999).

There are several types of artificial neural networks used in machine learning systems such as follows:

3.2.2.1 Multi Layer Perceptron neural network

Multi-Layer Perceptron neural network is the most popular type of artificial neural networks. Generally, MLP neural network applied the backpropagation learning method to tune its weights. Due to their excellent learning ability and nonlinear mapping capacity, the MLP neural network has been applied in SOC estimation. It is worth noting that, in SOC estimation, the relationship between the input and target is a nonlinear relation (WEIGERT et al., 2011).

According to (LINDA et al., 2009), the Multi-Layer Perceptron neural network is applied in SOC estimation using the recent history of voltage, current and the ambient temperature of a battery.

Section 2.3.1 details the ensemble of a few backpropagation Multi-Layer Perceptron neural networks which individual results are combined through bagging approach.

3.2.2.2 RBF neural network

The RBF neural network is a useful estimation methodology for systems where the main goal is to obtain more detailed information from an incomplete training set. The RBF neural network has been used in SOC estimation, according to (CHANG, 2012). Results achieved an adequate operation speed and estimation accuracy an, for this reason, meets the demands in practice. In (CHANG, 2012), the RBF neural network SOC estimation method uses the input data of the terminal voltage, discharging current, and temperature of the battery to estimate the SOC for $LiFePO4$ battery under different discharging conditions.

In order to eliminate the battery degradation's effect on the SOC estimation accuracy of the original trained model, (KANG et al., 2014) proposes a new Radial Basis Function Neural Network ($RBFNN$) model. Through simulations, the new model proved to have higher accuracy of the SOC estimation and good robustness against varying aging cycles, temperatures, and loading profiles.

3.2.2.3 Adaptive neural network

An adaptive neural network is built upon a feed-forward Multi-Layer Perceptron neural network with adaptive and recurrent feedback links from user-selected nodes. The connection topology may vary from one application to another, and the network weights are adaptively optimized using the recursive Levenberg-Marquardt algorithm (RLM) which allows the network to learn from the past (LIU et al., 2010).

Features extracted from sensor data of voltage, current, power, impedance, frequency, and temperature readings are used to estimate the internal impedance of a battery set containing lithium-ion cells that were cycle-life tested at 60 percent of the state of charge and temperature ($25C$ and $45C$) (LIU et al., 2010).

According to (REZVANIZANIANI et al., 2011), it is applied an Adaptive Neural Network ($AdNN$) for Remaining Useful Life (RUL) estimation of lithium-ion batteries based on the estimation of its internal impedance.

3.2.3 Fuzzy logic method

Fuzzy logic models are applied in complex and nonlinear problems. Therefore, they can be used to analyze data obtained by impedance spectroscopy and Coulomb counting methods. In these fuzzy logic models, the *ac* battery internal impedance and the voltage recovery are applied as input parameters (SALKIND et al., 1999). An interesting application of a fuzzy logic-based SOC estimation is in portable defibrillators (SINGH et al., 2006).

3.2.4 Support vector machine

In a classification problem, the support vector machine (SVM) methods aim to build hyperplanes to maximize the minimum distances between individuals from the training set classified into distinct categories. The SVM has also been applied for regression problem. In this case, a threshold is defined for each numerical attribute to divide the individuals according to the numerical attribute. The SVM achieves good results in the estimation of the SOC of lithium-ion battery since it is insensitive to small changes (HANSEN; WANG, 2005).

3.2.5 Kalman filter

According to (XU et al., 2012), the Kalman filter method provides accurate estimations of battery SOC. (YATSUI; BAI, 2011) presents a Kalman filter based SOC estimation method for lithium-ion batteries which effectiveness is validated through experimental results. It is proposed by (BARBARISI et al., 2006) an extended Kalman filter (EKF) to estimate the concentrations of the main chemical battery components by using the terminal current and voltage measurements. The concentrations of the main chemical battery components are related to the battery internal impedance which is proportional to battery SOC.

3.3 Hybrid methods

Direct measurements and machine learning methods approaches can be combined to leverage the performance metrics of the predictive models by using the strengths of both approaches (LIU et al., 2012).

3.3.1 Coulomb counting and EMF combination

The combination of direct measurement method (Coulomb counting method) during the discharge state with the battery EMF - electromotive force - measurement has been developed and implemented in an operational system (POP et al., 2009).

3.3.2 Coulomb counting and Kalman filter combination

According to (WANG et al., 2007), a new SOC estimation method, denoted as "Kalman Ah method", can be applied. This method applies the Kalman filter method to estimate the initial value used in the Coulomb counting method. Then, the Coulomb counting method is applied to adjust the SOC value along the further operational cycles. With this hybrid approach, the SOC estimation error decreased to 25 percent of the error when using Coulomb counting method.

3.3.3 Per-unit system and EKF combination

According to (KIM; CHO, 2011), EKF method combined with a per-unit (PU) system results in high accuracy estimation of lithium-ion battery SOC. According to this hybrid approach, the absolute values of the parameters in the equivalent circuit model in addition to the terminal voltage and current are converted into dimensionless values relative to a set of base value. Therefore, the converted values are applied to dynamic and measurement models in the EKF algorithm.

3.4 Li-ion battery state of charge (SOC) based on battery internal impedance

The methods described in the previous Sections aim to estimate the SOC. However, the SOC estimation is directly related to the battery internal impedance estimation, which is the scope of this study.

According to the definition of the SOC, the function which provides the battery state of charge based on battery internal impedance is obtained as follows:

$$SOC_t = SOC_{t_0} + \int_{t_0}^{t} (\frac{\eta \cdot I}{C_t}) \cdot d_t \tag{3.4}$$

Where:

- SOC_{t_0}: estimated SOC at time t_0, when the estimation process starts

- SOC_t: estimated SOC at time t

- η: current efficiency

- I: current - assumed to be positive when charging

- C_n: capacity of the battery at time t

In this study, the battery impedance is obtained through the ensembles of machine learning models (see Section 4.4) which can be applied in the determination of battery state of charge (SOC).

3.5 Features selection for Li-ion battery impedance estimation

Battery impedance, which decreases over the working time of a battery, is an important and direct indicator for estimating SOH and remaining useful life of the battery (FARMANN et al., 2015). In online or in-orbit applications, such as electric vehicles and satellites, capacity measurement or monitoring is difficult due to the lack of sensing data available during operational cycles (LIU et al., 2013). According to (SAHA; GOEBEL, 2007), to accurately estimate the battery internal impedance, EIS technique can be applied through offline tests under optimal measuring conditions and by using specialized and expensive equipment (DALAL et al., 2011). However, during its operational life, the battery cannot be removed very often to conduct EIS tests. Also, these tests are expensive.

Due to the need of applying operational data in the estimation of the battery internal impedance, operating direct parameters, such as voltage, current, and temperature, could be used in the development of an estimation model. Although, in practical applications, such characteristics are controlled to meet the load requirements of an associated circuit and cannot represent battery aging (PARVIZ; MOIN, 2011). For this reason, indirect features shall be used to characterizes the battery operational cycles.

According to (ZHANG; LEE, 2011), six features characterize each $C - D$ cycle and can be applied in machine learning systems modeling to estimate battery internal impedance, as described as follows:

3.5.1 F1 and F2 - Time intervals extracted from CC/CV charge step

The first two features are charge related and extracted from the CC/CV charge step. According to (EDDAHECH et al., 2014), the CC capacity decreases with battery aging. In each charging step, the battery is first charged in CC (continuous current) mode and then in CV (continuous voltage) mode. The fixed cutoff voltage and current cannot provide direct degradation information for battery internal impedance estimation. However, the time intervals between the nominal voltage and the cutoff voltage ($F1$) and between the nominal current and the cutoff current ($F2$) can be applied as features.

3.5.2 F3 - Time interval between two predefined discharge voltages

In the initial operational cycles, lithium-ion batteries of new cell phones/laptops have the maximum operating time. However, the operating time (discharging time period) after subsequent full charge becomes shorter and shorter. Also, there is a certain relationship between the discharging period and the capacity of lithium-ion batteries. Therefore, the time intervals of equal discharging voltage differences ($F3$) can be applied as features in the SOC estimation (LIU et al., 2013).

The $TIEDVD$ parameter is defined as the time interval corresponding to a certain discharging voltage difference according to Equation 3.5:

$$TIEDVD_k = (t_{V\,max} - t_{V\,min})_k \tag{3.5}$$

Where:

- $TIEDVD_k$: time difference between two predefined discharging voltage at cycle k
- $t_{V\,max}$: time during the discharge cycle when battery discharge voltage achieves a predefined maximum value at cycle k
- $t_{V\,min}$: time during the discharge cycle when battery discharge voltage achieves a predefined minimum value at cycle k

3.5.3 F4 and F5 - Average temperatures during charge and discharge

The body temperature of a Li-ion battery affects its thermal behavior and its capacity and resistance. Thus, the fourth and fifth degradation features are average temperatures during charge and discharge step, respectively (ONDA et al., 2006).

The fourth feature ($F4$) is extracted from the charge step and corresponds to the average temperature during the time interval $F1$. Similarly, the fifth feature ($F5$) is extracted from the discharge step and corresponds to the average temperature during the time interval $F3$.

3.5.4 F6 - Cutoff voltage in discharge step

The discharge cutoff voltage is related to depth of discharge (DoD) which, according to (SATO, 2001), impacts the battery performance. Therefore, the discharge cutoff voltage is considered as the sixth feature in our model ($F6$).

Next chapter will detail how these features were extracted from the Nasa's dataset.

4 DATA PREPARATION AND MODELING FOR LI-ION BATTERY IMPEDANCE ESTIMATION

In this Chapter, the battery dataset preparation and modeling is described in details. This chapter defines a data manipulation and analysis process which transforms the raw battery testing data into a structured dataset. The obtained dataset is divided into training, validation and testing sets and used by machine learning algorithms to build two ensembles of machine learning models based on MLP neural network bagging (Section 2.3.1) and gradient tree boosting (Section 2.3.2).

After performing tests to compare both models, their results are compared with the following model applied to the same battery dataset and tested using the same testing set: A Comparative Analysis of Techniques for Electric Vehicle Battery Prognostics and Health Management (PHM)(REZVANIZANIANI et al., 2011). The Adaptive Neural Network proposed in this paper is according to Section 3.2.2.3 and its obtained performance is described in Section 5.3.3.

4.1 Li-ion battery testing set information

In the present study, machine learning models in Li-ion battery internal impedance estimation are developed upon data provided by National Aeronautics and Space Administration (NASA) Ames Prognostics Center of Excellence (SAHA; GOEBEL, 2007).

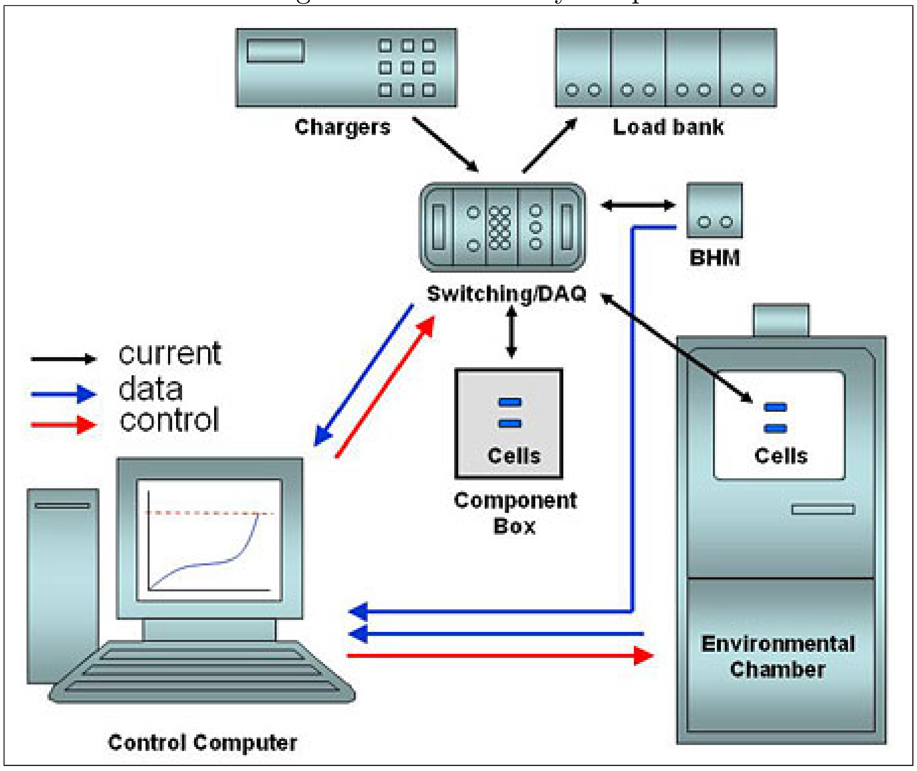
The laboratory setup and data recording were conducted by National Aeronautics and Space Administration (NASA) Ames Prognostics Center of Excellence (SAHA; GOEBEL, 2007). According to NASA experiment, the laboratory setup is according Section 4.1.1 and the experimental data organized according to Section 4.1.2.

4.1.1 Laboratory setup

The laboratory setup contains an operational battery (containing a set of Li-ion cells), loads, chargers and an assembly of devices for battery health monitoring (BHM) containing sensors of temperature, voltage and current, switches, data acquisition system and a computer for control and analysis. Figure 4.1 details the full stack.

The Li-ion cells are submitted to charge and discharge cycles under different environmental conditions and load set by the environmental chamber and electronic load controller, respectively. In addition, periodically, EIS measurements (which

Figure 4.1 - Laboratory setup



Source: (SAHA; GOEBEL, 2007)

obtain the battery internal impedance for a certain range of *ac* frequencies) are collected through BHM module. The switches alternate the operational battery cycles between charging, discharging and EIS cycles (SAHA; GOEBEL, 2007).

4.1.2 Battery testing set information

The Li-ion batteries operate under different operational profiles (charge, discharge and impedance) at ambient temperatures of $4, 24$ and 44°C (SAHA; GOEBEL, 2007):

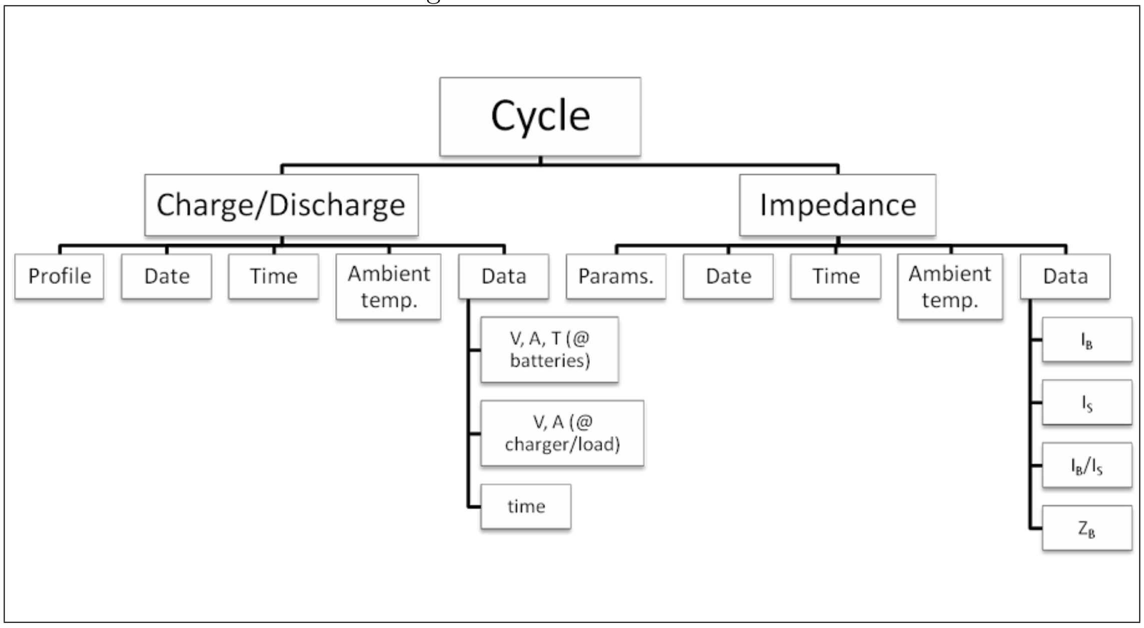
a) Charge step: charging was carried out in a constant current (CC) mode at $1.5A$ until the battery voltage reached 4.2V and then continued in a constant voltage (CV) mode until the charge current dropped to $0.02A$

b) Discharge step: discharging was conducted in CC mode until the discharge voltage reached a predefined cutoff voltage. Fixed and variable load currents at $1, 2,$ and $4A$ were used and the discharge runs were stopped at $2V, 2.2V, 2.5V$ or $2.7V$

c) Impedance measurement: measurement was performed through an electrochemical impedance spectroscopy (EIS) frequency sweep from $0.1Hz$ to $5kHz$

Figure 4.2 details the battery data structure of the operational profiles.

Figure 4.2 - Data structure



Source: (SAHA; GOEBEL, 2007)

4.1.3 Li-ion battery impedance measurement rectifier

To eliminate the noise generated by time-varying current passing through an electrochemical cell or battery due to load fluctuation, a filtering approach or an electronic cancellation technique shall be applied. In the laboratory setup described above, it was used an electronic device.

The time-varying current flowing through the circuit detailed in Section 4.1.1 excites a magnetically-coupled *ac* current probe which produces an opposite signal that is amplified and injected into the circuit, canceling the oscillation of the original signal.

4.2 Data preparation for Li-ion battery impedance estimation

The data detailed in Figure 4.2 is structured in two distinct tables. The first table (denominated as external temperature, voltage, and current experimental data) contains external temperature, voltage and current measured directly after the battery termination along the $C - D$ battery cycles. The second table (denominated as electrochemical impedance spectroscopy experimental data) is obtained through the spectroscopy experiments and comprises with the battery internal impedance along the same $C - D$ cycles.

4.2.1 External temperature, voltage and current experimental data

A few snapshots of the first table are according to 4.1.

Table 4.1 - External temperature, voltage and current experimental data

profile	year	mon	day	hour	min	sec	temp	voltage	current	cycle
discharge	2010	9	3	12	10	9.43	6.31	3.84	0.00	1
discharge	2010	9	3	12	10	25.7	6.36	3.31	-1.99	1
discharge	2010	9	3	12	10	26.2	6.44	3.28	-1.99	1

This table contains the time series of the following parameters measured directly after the battery termination along the C-D battery cycles:

- external temperature (in F)
- voltage (in V)
- current (in A)

NOTE:

In this study, during the discharge process, they were only considered the load current of $2.0A$ and the snapshots between $3.8V$ and $3.0V$.

To obtain the features detailed in Section 4.3, the snapshots are aggregated by charge and discharge cycle. However, prior to the features computation, outliers shall be excluded. These outlier records do not comprise with one of the following requirements:

- During the charge step carried out in a constant current (CC) mode, the minimum battery voltage shall be $3.8V$ and the charging current shall be equal to $1.5A$ until the battery voltage maximum limit which shall not exceed $4.2V$

- During the charge step carried out in a constant voltage (CV) mode, the charging current shall be equal to $4.2V$ until the battery current minimum limit which shall not dropped below to $0.02A$

4.2.2 Electrochemical impedance spectroscopy experimental data

A few snapshots of the second table are according to 4.2.

Table 4.2 - Electrochemical impedance spectroscopy experimental data

profile	year	mon	day	hour	min	sec	impedance	rectified imp	cycle
EIS	2010	9	3	12	10	9.43	0.05-0.42i	0.23-0.01i	1
EIS	2010	9	3	12	10	25.7	0.17-0.02i	0.22-0.01i	1
EIS	2010	9	3	12	10	26.2	-0.02-0.02i	0.22-0.01i	1

NOTE:

They were only considered the electrochemical impedance spectroscopy experimental results with a frequency of $2.0Hz$ and the rectified battery impedance according to Section

4.3 Specific features selection for Li-ion battery impedance estimation

The laboratory setup and data recording conducted by National Aeronautics and Space Administration (NASA) Ames Prognostics Center of Excellence(SAHA; GOEBEL, 2007) resulted in testing data containing data from the operation cycles of 134 rechargeable lithium-ion batteries organized in 34 battery data sets. Each battery dataset contains the test data according to Figure 4.2. The testing data applied in this study is according to (SAHA; GOEBEL, 2007). In this study, a dataset structured as detailed in 4.2 is used in the computation of the features to be applied in the modeling stage.

For each $C - D$ (charge and discharge) cycle, the following features were extracted exactly as proposed by (ZHANG; LEE, 2011) (according to Section 3.5):

- F1: during charge cycle, time interval between the nominal voltage and the cutoff voltage
- F2: during charge cycle, time interval between the nominal current and the cutoff current

Table 4.3 - Historical set - impedance real component

cycle	F1	F2	F3	F4	F5	F6	imp_re
1	9966.407	6422.609	472.313	27.0074	26.3008	2.4556	0.06175
2	10226.375	6627.891	472.125	25.6742	26.5323	2.6321	0.05989
3	10635.968	6528.063	472.344	25.6754	26.3325	2.5010	0.05919

- F3: during discharge cycle, time interval between two predefined voltages
- F4: average temperature during the time interval F1
- F5: average temperature during the time interval F2
- F6: during discharge cycle, cutoff voltage

The historical set applied in the machine learning systems modeling includes the six features (F1, F2, F3, F4, F5, and F6) and the label attribute which corresponds to the rectified battery impedance.

In order to obtain the rectified battery impedance correspondent to a specific $C-D$ cycle of the external temperature, voltage and current experimental data table (4.1), the mean real and imaginary components of the Li-ion battery internal impedance were obtained for every $C-D$ cycle of electrochemical impedance spectroscopy experimental data table (4.2). Thus, the mean Li-ion battery internal impedance correspondent to a specific $C-D$ cycle is the simple average of the Li-ion battery internal impedance along all the timestamps of the cycle. During each timestamp, the mean Li-ion battery internal impedance is calculated according to Equation 4.1.

$$\text{impedance} = \sqrt{(real + imaginary)^2} \tag{4.1}$$

Where:

- *real*: real component of the Li-ion battery internal impedance
- *imaginary*: imaginary component of the Li-ion battery internal impedance

Tables 4.3 and 4.4 are extractions of the historical sets corresponding to the real and imaginary components which contain 600 $C-D$ training cycles.

Table 4.4 - Historical set - impedance imaginary component

cycle	F1	F2	F3	F4	F5	F6	imp_img
1	9966.407	6422.609	472.313	27.0074	26.3008	2.4556	-0.00096
2	10226.375	6627.891	472.125	25.6742	26.5323	2.6321	-0.00112
3	10635.968	6528.063	472.344	25.6754	26.3325	2.5010	-0.00105

4.4 Li-ion battery impedance data modeling

After the data preparation according (ZHANG; LEE, 2011), two ensembles of machine learning models (MLP neural network bagging - Section 2.3.1 and gradient tree boosting - Section 2.3.2) were developed in order to estimate the mean Li-ion battery internal impedance.

To test the two ensembles of machine learning models and compare the obtained results with the benchmark approach, the database was split into a training set (applied in the development of the ensembles of models) and a testing set (applied in the determination of the performance achieved by the ensembles of models). To consider the same train and test $C-D$ cycles as were considered in the benchmark analysis and allow the comparison between the results, the training set was formed by the $C-D$ cycles from 1 to 380. The testing set used in the evaluation of the two ensembles of models developed herein is the same as the benchmark testing set and comprises with the $C-D$ cycles from 380 to 600.

After segregating the database in training and testing sets, the training set containing the $C-D$ cycles from 1 to 380 is divided into training and validation sets to allow the evolution of the ensembles of models through some training cycles. During each training cycle, the training set is split randomly into 70-30. The larger split is used to fit the ensemble models while the smaller split is segregated and applied after the modeling in the validation process.

Therefore, within a training cycle, the hyperparameters which tune the ensembles architectures and the configurations of the base learners are adjusted. From the training set, each training cycle generates a modified ensemble of base learners. To compare the generated ensembles of regression models, during a training cycle, for a specific combination of the hyperparameters, the specific ensemble of models is applied to the validation set.

The model application allows the estimation of this mean Li-ion battery internal impedance that can be compared with the real value. Applying the $RMSE$ performance metric in the validation set, the deviations between the estimated and real values are aggregated, and the quality of the ensemble of models is obtained. At each validation iteration, the training set is randomly split into 70-30 subsets (training and validation sets) and the ensemble of models, tuned with a specific set of hyperparameters, is validated as described in Section . After the validation cycles, the best hyperparameters configuration which tuned the best ensemble of models can be selected to be applied in the test set.

4.5 MLP neural network bagging with monotonicity constraints model

In this ensemble of models, a base learner is a MLP neural network with monotonicity constraints which implements one hidden-layer that can enforce or not monotonic relations on designated input variables. Each training cycle applies 10 or 20 ensemble members to fit, enforce or not the monotonicity and applies 1, 2, 3, 4, 5, 6, 7, 8, 9 or 10 hidden nodes in the hidden layer.

The ensemble members to fit in each training cycle are obtained according to Section 2.2.1. Each ensemble member contains a random subset of 70 percent of the training set and, after generating the 10 or 20 regression models (depending on the number of ensemble members to fit), the result corresponds to the mean value obtained through the application of all regression models.

These training cycles with different combinations result into different MLP neural networks with monotonicity constraints. These different combinations of the hyperparameters are described in Table 4.5.

Table 4.5 - MLP neural network with monotonicity constraints hyperparameters

hyperparameter	description	possible values
hidden nodes	number of hidden nodes in the first hidden layer	1 to 10
ensemble learners	number of ensemble members to fit	10, 20
monotonicity	column indices of covariates for which the monotonicity constraint should hold	0 or 1
bag	logical variable indicating whether or not bootstrap aggregation (bagging) is used	TRUE
iter.max	maximum number of iterations of the optimization algorithm	500

Each combination of the hyperparameters tune a specific ensemble of models which has a performance metric obtained through the application of $RMSE$ in the validation set. Figure 4.3 details the root mean square error (horizontal axis) calculated according to Equation 4.1 of the 16 best bagging ensembles of MLP models which hyperparameters are according to Table 4.5. Each one of the 16 best bagging ensembles is represented in the Figure 4.3 by a set of three colored bars. Each hyperparameter configuration of the 16 best bagging ensembles of MLP models is represented by a specific color representing a value in the vertical axis as follows:

- Blue bars: present the number of base learners applied in each one of the ensembles
- Orange bars: present the number of nodes (neurons) applied in the hidden layer of the MLP base learners which form the ensembles
- Gray bars: present the usage or not of the monotonicity criteria in the updating process of the MLP base learners weights ('0' means that the monotonicity criteria was not applied while '1' means that the monotonicity criteria was applied)

Figure 4.3 starts with the hyperparameters configurations which resulted in the ensemble methods with the lower root mean square error when applied in the validation set. The ensemble of MLP models that achieved the lower root mean square error was tuned with the following hyperparameters: 20 ensemble learners, 6 neurons in the hidden layer and no monotonicity forcing.

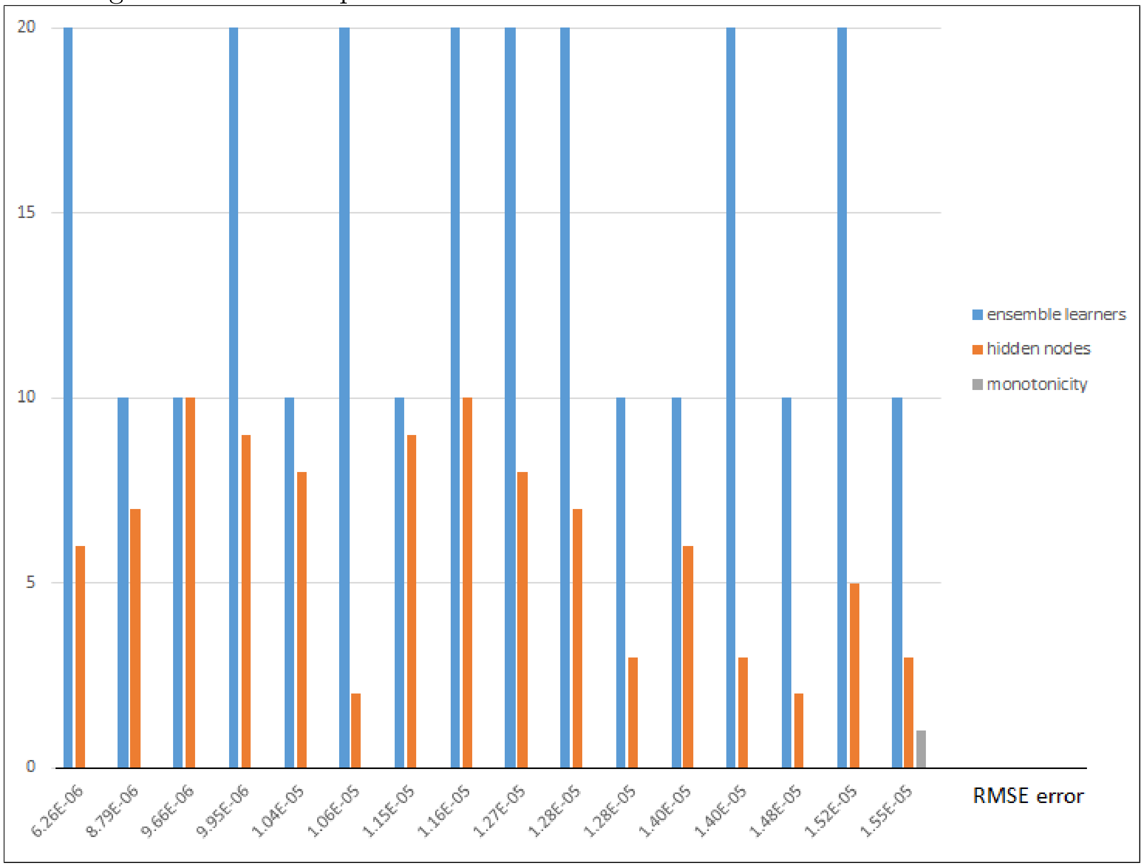
In the other hand, the ensemble of MLP models that achieved the higher root mean square error was tuned with the following hyperparameters: 20 ensemble learners, 10 neurons in the hidden layer and monotonicity forcing.

4.6 Gradient tree boosting model

In this ensemble of models, a base learner is a decision tree with a maximum depth of 4 or 8. In the boosting ensemble of the base learners, the learning rate, applied in the computation of the next learner, varies between 0.1 to 1.6 with a step of 0.1. Finally, to reduce the variance, different subsets of the features can be used by the base learners. In the boosting process, 80 percent of the features were randomly used by the learners or all the features were used by them.

These different combinations of the hyperparameters result into different gradient boosting ensembles of models, as described in Table 4.6.

Figure 4.3 - Mean square error of 16 best ensembles of MLP neural networks



The hyperparameter η has an important role in the gradient boosting modeling since it controls the learning rate. This parameter scales the contribution of each tree by a factor between 0 and 1. It is used to prevent overfitting by making the boosting process more conservative. A lower learning rate - η - increases the robustness to overfitting but also the computing time.

Each combination of the hyperparameters tune a specific ensemble of models which has a performance metric obtained through the application of $RMSE$ in the validation set according to Section 2.4.3. Figure 4.4 details the root mean square error (horizontal axis) calculated according to Equation 4.1 of the 16 best *xgBoost* ensembles of models which hyperparameters are according to Table 4.6. Each one of the 16 best boosting ensembles is represented in the Figure 4.4 by a set of three colored bars. Each hyperparameter configuration of the 16 best *xgBoost* ensembles of models is represented by a specific color representing a value in the vertical axis as follows:

Table 4.6 - Gradient boosting hyperparameters

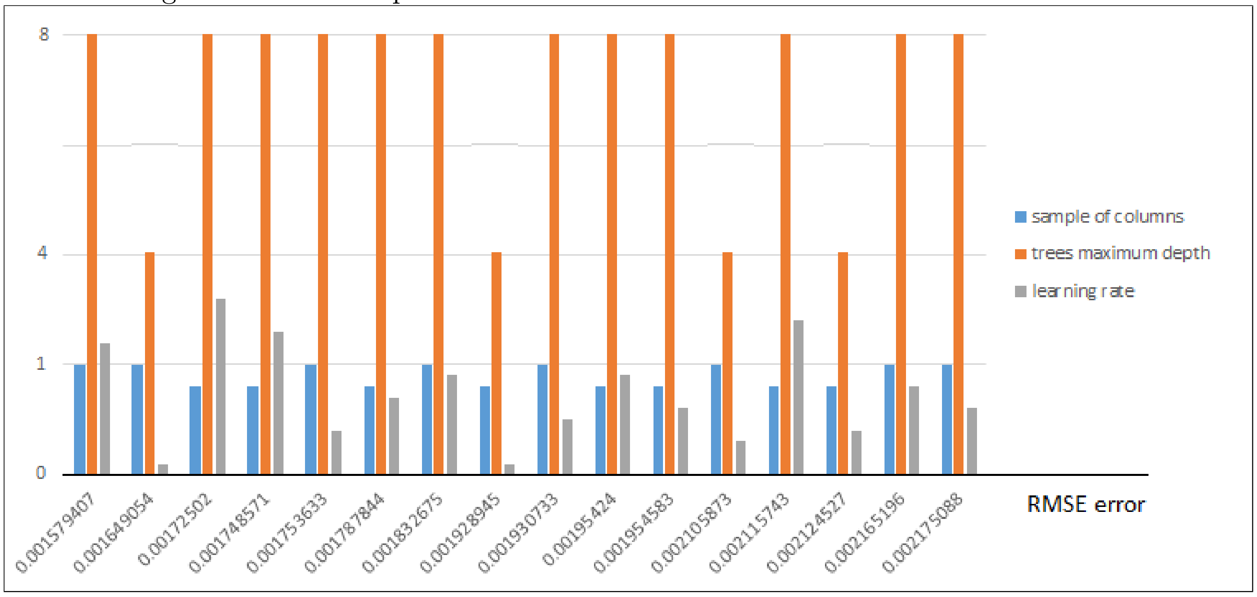
hyperparameter	description	possible values
objective	objective function	reg:linear
max_depth	maximum depth of a tree	4 or 8
η	control the learning rate: scale contribution of each tree by a factor of 0 <eta <1	0.10 to 1.60
column_sample	subsample ratio of columns when constructing each tree	1.0 or 0.80
instance_sample	subsample ratio of the training instance. 0.5 means that xgboost randomly collected half of the data to grow trees and this will prevent overfitting	1.0
evaluation_metric	evaluation metric per validation cycle	$RMSE$
validation cycles	the max number of validation cycles	200

- Blue bars: present the rate of features (from '0' to '1') selected randomly by each base learner. '1' means that all features are applied
- Orange bars: present the maximum depth of the decision trees which form the *xgBoost* ensembles of models
- Gray bars: present the learning rate - η - of the *xgBoost* ensembles of models

Figure 4.4 starts with the hyperparameters configurations which resulted in the ensemble methods with the lower root mean square error when applied in the validation set. The ensemble of decision tree models that achieved the lower root mean square error was tuned with the following hyperparameters: decision trees maximum depth of 8 levels, no column sample (all the base learners applies all the features) and learning rate of 1.2 in the development of the consecutive learners.

In the other hand, the ensemble of decision tree models that achieved the higher root mean square error was tuned with the following hyperparameters: decision trees maximum depth of 4 levels, column sample of 80 percent (all the base learners applies randomly 80 percent of all the features) and learning rate of 0.4 in the development of the consecutive learners.

Figure 4.4 - Mean square error of 16 best ensembles of decision trees



5 COMPUTER EXPERIMENTS AND RESULTS FOR LI-ION BATTERY IMPEDANCE ESTIMATION

In this Chapter, the ensemble of models described in Chapter 4 is tested in the C-D testing cycles from 380 to 600. After performing tests to comparing both models, their results are compared with the following model applied to the same battery dataset and tested using the same testing set: A Comparative Analysis of Techniques for Electric Vehicle Battery Prognostics and Health Management (PHM)(REZVANIZANIANI et al., 2011). The Adaptive Neural Network proposed in this paper is according to Section 3.2.2.3 and its obtained performance is described in Section 5.3.3.

5.1 MLP neural network bagging with monotonicity constraints test

After developing, ensembles of MLP neural network and validating them with the validation set (randomly selected as 30 percent of the training set), the best ensembles of models are detailed in Table 5.1.

Table 5.1 - Best ensembles of MLP neural network models

ensemble	hidden nodes	ensemble learners	monotonicity	bag	iter.max
1	6	20	0	TRUE	500
2	7	10	0	TRUE	500
3	10	10	0	TRUE	500

To obtain a more accurate performance metric, the three MLP ensembles of models are validated through cross-validation methodology according to Section 4.4. Thus, instead of segregating the training set randomly in 70-30 subsets, the $C-D$ training cycles from 1 to 380 are divided into five folds. During the first iteration, the first fold (containing 76 $C-D$ cycles) is used to validate the ensemble of models developed with the last four folds (containing 304 $C-D$ cycles). The $RMSE$ performance metric of the first iteration is obtained through the application of the ensembles of models to the validation set according to Section 2.4.3. This process is repeated five times. At each iteration, a specific fold is used as validation set and the remaining folds as training set. Finally, five $RMSE$ performance metrics are obtained and the mean $RMSE$ metric is considered as the $RMSE$ cross-validation performance metric. Table 5.2 enumerates the $RMSE$ metrics obtained through the each validation cycle and the mean $RMSE$ cross-validation performance metric.

Table 5.2 - Performance metrics of best MLP bagging ensembles at cross validation iterations

iteration	ensemble 1	ensemble 2	ensemble 3
1	0.0049	0.0051	0.0053
2	0.0030	0.0027	0.0030
3	0.0061	0.0056	0.0054
4	0.0020	0.0022	0.0013
5	0.0023	0.0032	0.0023
MEAN	0.0037	0.0038	0.0035

The best three ensembles according to Table 5.1 are tested through $C-D$ testing cycles from 380 to 600. For each one of the $C-D$ testing cycles, the estimated battery internal impedance is compared with the real value in ω. Figure 5.1 shows the estimated values of the battery internal impedance according to each one of the ensembles detailed in Table 5.1.

The differences between the estimated and real mean Li-ion battery internal impedances during each one of the $C-D$ testing cycles are according to Figure 5.2 and the $RMSE$ error of the three ensembles when applied on the testing set are according to Table 5.7.

5.2 Gradient tree boosting test

5.2.1 Gradient tree boosting optimal ensembles

After developing ensembles of decision trees through gradient tree boosting and validating them with the validation set (randomly selected as 30 percent of the training set), the best ensembles of models are detailed in Table 5.3.

Table 5.3 - Best gradient tree boosting ensembles

ensemble	max depth	columns sample	η - learning rate	instances sample	evaluation metric
1	8	1.0	1.2	1	200
2	4	1.0	0.1	1	200
3	8	0.8	1.6	1	200

In order to obtain a more accurate performance metric, the three *xgBoost* ensembles of models are validated through cross-validation methodology according to Section

Table 5.4 - Performance metrics of best gradient tree boosting ensembles at cross validation iterations

iteration	ensemble 1	ensemble 2	ensemble 3
1	0.0050	0.0050	0.0030
2	0.0054	0.0035	0.0065
3	0.0045	0.0044	0.0089
4	0.0039	0.0048	0.0057
5	0.0022	0.0059	0.0041
MEAN	0.0042	0.0047	0.0057

4.4. Thus, instead of segregating the training set randomly in 70-30 subsets, the $C-D$ training cycles from 1 to 380 are divided into five folds. During the first iteration, the first fold (containing 76 $C-D$ cycles) is used to validate the ensemble of models developed with the last four folds (containing 304 $C-D$ cycles). The $RMSE$ performance metric of the first iteration is obtained through the application of the ensembles of models to the validation set according to Section 2.4.3. This process is repeated five times. At each iteration, a specific fold is used as validation set and the remaining folds as training set. Finally, five $RMSE$ performance metrics are obtained and the mean $RMSE$ metric is considered as the $RMSE$ cross-validation performance metric. Table 5.4 enumerates the $RMSE$ metrics obtained through the each validation cycle and the mean $RMSE$ cross-validation performance metric.

The best three ensembles according to Table 5.1 are tested through $C-D$ testing cycles from 380 to 600. Figure 5.3 shows the estimated and real values of the Li-ion battery internal impedance according to each one of the ensembles.

The differences between the estimated and real battery internal impedances during each one of the $C-D$ testing cycles are according to Figure 5.4 and the $RMSE$ error of the three ensembles when applied on the testing set are according to Table 5.7.

5.2.2 Gradient tree boosting features importance

According to Section 2.3.2.3, a specific feature of the data set has distinct importance values in the decision trees which form the gradient tree boosting ensemble. For the three best ensembles according to Table 5.3, the average importance values of the features in all their base learners are indicated in Table 5.5.

The features importance in the three ensembles can be averaged. Thus, the mean

Table 5.5 - Features importance in the best three gradient tree boosting ensembles

feature	ensemble 1	ensemble 2	ensemble 3
F1	0.044	0.147	0.035
F2	0.021	0.099	0.344
F3	0.693	0.316	0.249
F4	0.100	0.152	0.063
F5	0.020	0.116	0.006
F6	0.123	0.170	0.304

importance of the six features in the three ensembles is according to Table 5.6 which is sorted by importance.

Table 5.6 - Mean features importance in the best three gradient tree boosting ensembles

feature	mean importance
F3	0.419
F6	0.199
F2	0.155
F4	0.105
F1	0.075
F5	0.047

According to Table 5.6, the most relevant feature for the estimation of the battery internal impedance through gradient tree boosting ensembles of models is the time interval between two predefined discharge voltages (F3). Therefore, as empirically noted according to Section 3.5, the discharge period is indeed a reliable indicator of the battery degradation.

In another hand, the average body temperature of the battery during discharge is the less relevant feature for the estimation of its internal impedance through gradient tree boosting ensembles of models. Thus, the laboratory setup can be simplified, and the temperature sensing can be removed from it without a significant impact in the battery internal impedance estimation.

5.3 Li-ion battery impedance model test

According to $RMSE$ (see Section 2.4.3), the MLP ensemble of models with the higher performance on the testing set has the configuration of 20 ensemble members to fit and 6 hidden nodes on the hidden layer while the *xgBoost* ensemble of models with the higher performance has (η=0.10 and a features sampling ratio of 1.00).

Table 5.7 enumerates the $RMSE$ performance metrics obtained through the application of the best ensembles of models based on MLP bagging and *xgBoost* in the testing set ($C-D$ cycles from 380 to 600).

Table 5.7 - RMSE error of best ensembles

ensemble	MLP	*xgBoost*
1	0.0019	0.0018
2	0.0023	0.0017
3	0.0024	0.0029

According to Table 5.7, the ensemble of models that achieved the lower $RMSE$ error when applied in the $C-D$ testing cycles from 380 to 600 is the second *xgBoost* ensemble of models which has the following hyperparameters configuration: decision trees maximum depth of 8 levels, no column sample (all the base learners applies all the features) and learning rate of 0.1 in the development of the consecutive learners.

In the other hand, the third *xgBoost* ensemble of models achieved the higher root mean square error when tuned with the following hyperparameters: decision trees maximum depth of 8 levels, column sample with 0.8 rate (a subset of 80 percent of the features is randomly chosen by the base learners) and learning rate of 1.6 in the development of the consecutive learners.

As described in Section 4.6, a high learning rate applied in the iterative boosting process which forms the *xgBoost* ensemble of models can lead to overfitting. The third *xgBoost* ensemble of models is an example of such phenomenon since it achieved a low $RMSE$ error during the validation process but a high $RMSE$ error during the testing process.

In general, the *xgBoost* ensembles of models achieved a lower $RMSE$ error when compared to MLP ensembles of models. In Section 5.3.2, the statistical similarity between the ensembles is calculated. In addition, Section 5.3.2 also compares the obtained ensembles of base learners with benchmark machine learning models, as mentioned in Section 1.2.

The following Sections describe the training and testing stages of the $AdNN$ ((REZVANIZANIANI et al., 2011)) model development. This machine learning approaches apply only the own target variable.

5.3.1 Li-ion battery impedance AdNN model

According to Section 3.2.2.3, the $AdNN$ training process applies the last five instances of the target variable as input of a neural network. Therefore, with each set of six instances, a neural network is trained. This training process repeats for the $C - D$ training cycles from 1 to 380 and during each iteration, the $AdNN$ model is updated. After the training iterations, the $AdNN$ model begins to predict the Li-ion battery impedance for the $C - D$ testing cycles from 380 to 600 testing data (REZVANIZANIANI et al., 2011).

5.3.2 Student's t-test application

According to Student's t-test technique (see Section 2.5), in order to check the statistic difference between the performance metrics of the the MLP and *xgBoost* ensembles of models when compared with each other and with the benchmark model, the Z value obtained through the Student's t-test formula shall be above a positive threshold value and below a negative threshold value. Herein, the adopted positive threshold value is of 0.05 while the adopted negative threshold value is of -0.05.

In the Student's t-test described herein, there were considered the best three ensembles of MLP and *xgBoost* models (according to Tables 5.1 and 5.3), respectively.

Therefore, the two samples containing three performance errors (each one obtained through the application of MLP bagging and *xgBoost* ensembles of models on the testing set) can be compared with each other and with the performance error obtained through the application of the benchmark model by using the Student's t-test formula (2.15).

Table 5.7 enumerates the three performance errors obtained through the application of best three MLP bagging and *xgBoost* ensembles of models on the testing set.

In order to calculate the Z value of the statistic difference of MLP bagging and *xgBoost* performance errors compared with each other and with the benchmark model performance error, the following assumptions were made:

- For the comparison between the MLP bagging and *xgBoost* performance metrics: σ is the population standard deviation of the data
- For the comparison of the MLP bagging and *xgBoost* performance metrics

with the benchmark model performance metric: σ is the sample standard deviation of the data

Table 5.8 enumerates the Z value of the statistic difference of MLP bagging and *xgBoost* performance metrics compared with each other and with the benchmark model performance metric.

Table 5.8 - Student's t-test analysis

Z value	MLP model	*xgBoost* model	AdNN model (REZV., 2017)
MLP model	-	0,17	-13,66
xgBoost model	-0,44	-	-5,60

5.3.3 Li-ion battery impedance model comparison

Considering the Z value of Table 5.8, the $RMSE$ error obtained in the estimation of the mean Li-ion battery internal impedance through MLP bagging and *xgBoost* ensembles of models are compared with the $RMSE$ error of the benchmark model described in Section 1.2: Adaptive Neural Network ($AdNN$)(REZVANIZANIANI et al., 2011).

The $RMSE$ mean error of the performance metric samples obtained through the application of the ensembles of regression models on the battery data set provided by National Aeronautics and Space Administration (NASA) (SAHA; GOEBEL, 2007) are according to Table 5.9.

Table 5.9 - RMSE error of the ensembles of regression models application

MLP	*xgBoost*	AdNN model (REZV., 2017)
0.0022	0.0021	0.0043

According to Table 5.8 and considering the threshold of $Z = 0.05$, the performance of the ensembles of models proposed herein and the benchmark models are distinct compared to each other.

The Gradient Tree Boosting (*xgBoost*) ensemble of models achieved the lower $RMSE$ error and, for this reason, is considered the most suitable in the Li-ion battery impedance estimation.

Figure 5.1 - Estimated and real battery internal impedance values through application of MLP bagging ensembles of models in the testing cycles

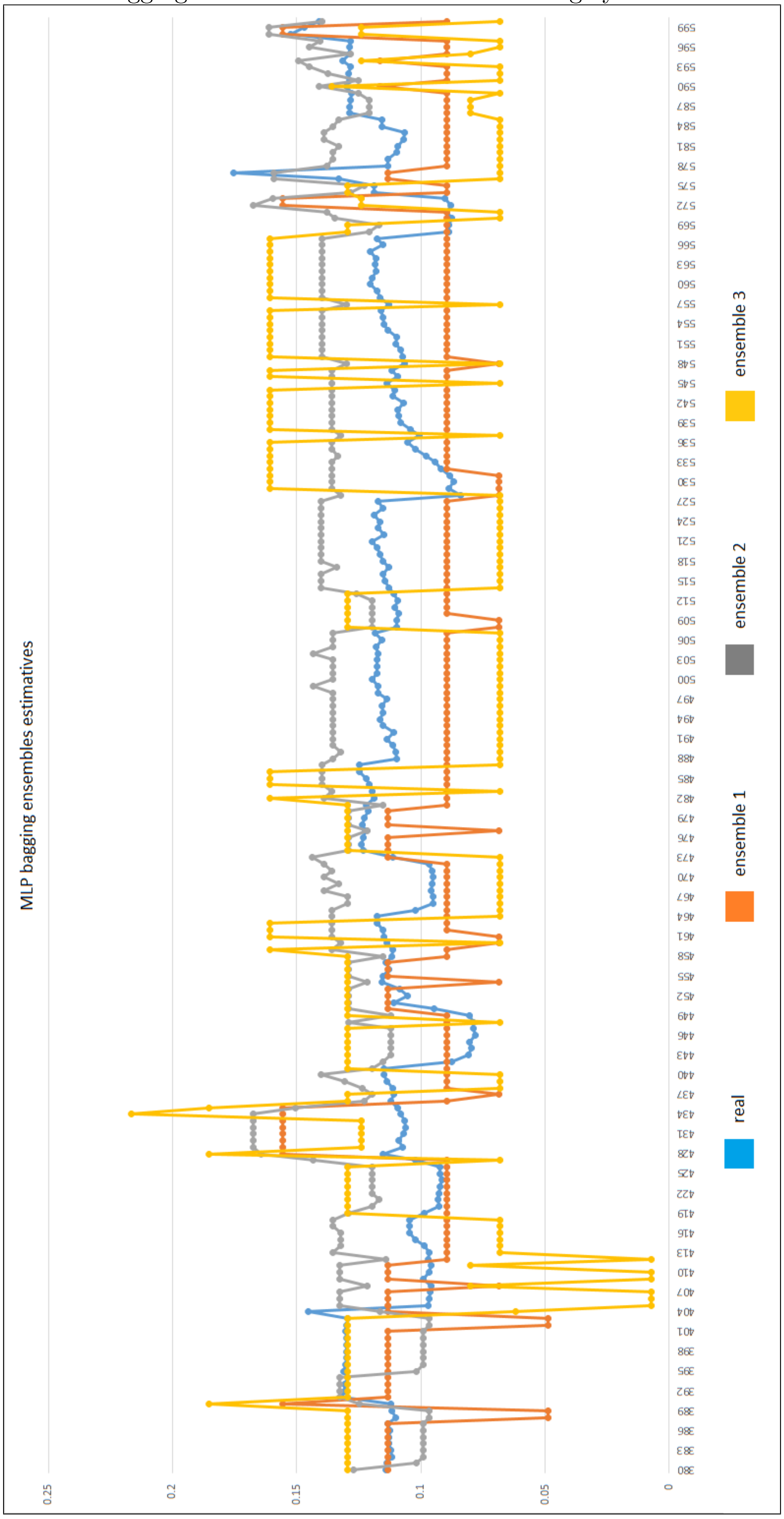


Figure 5.2 - Differences between estimated and real battery internal impedance values through application of MLP bagging ensembles of models in the testing cycles

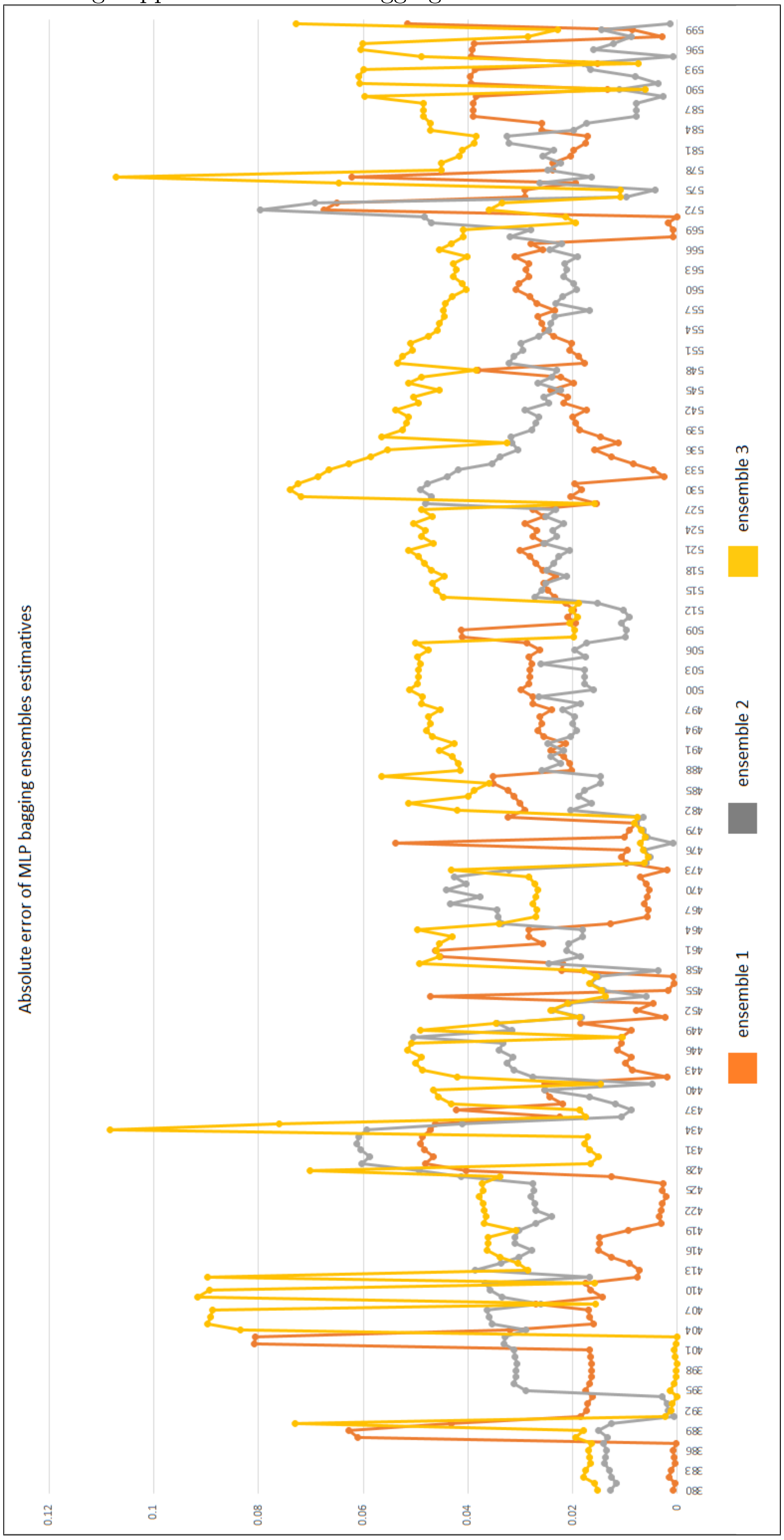


Figure 5.3 - Estimated and real battery internal impedance values through application of xgBoost ensembles of models in the testing cycles

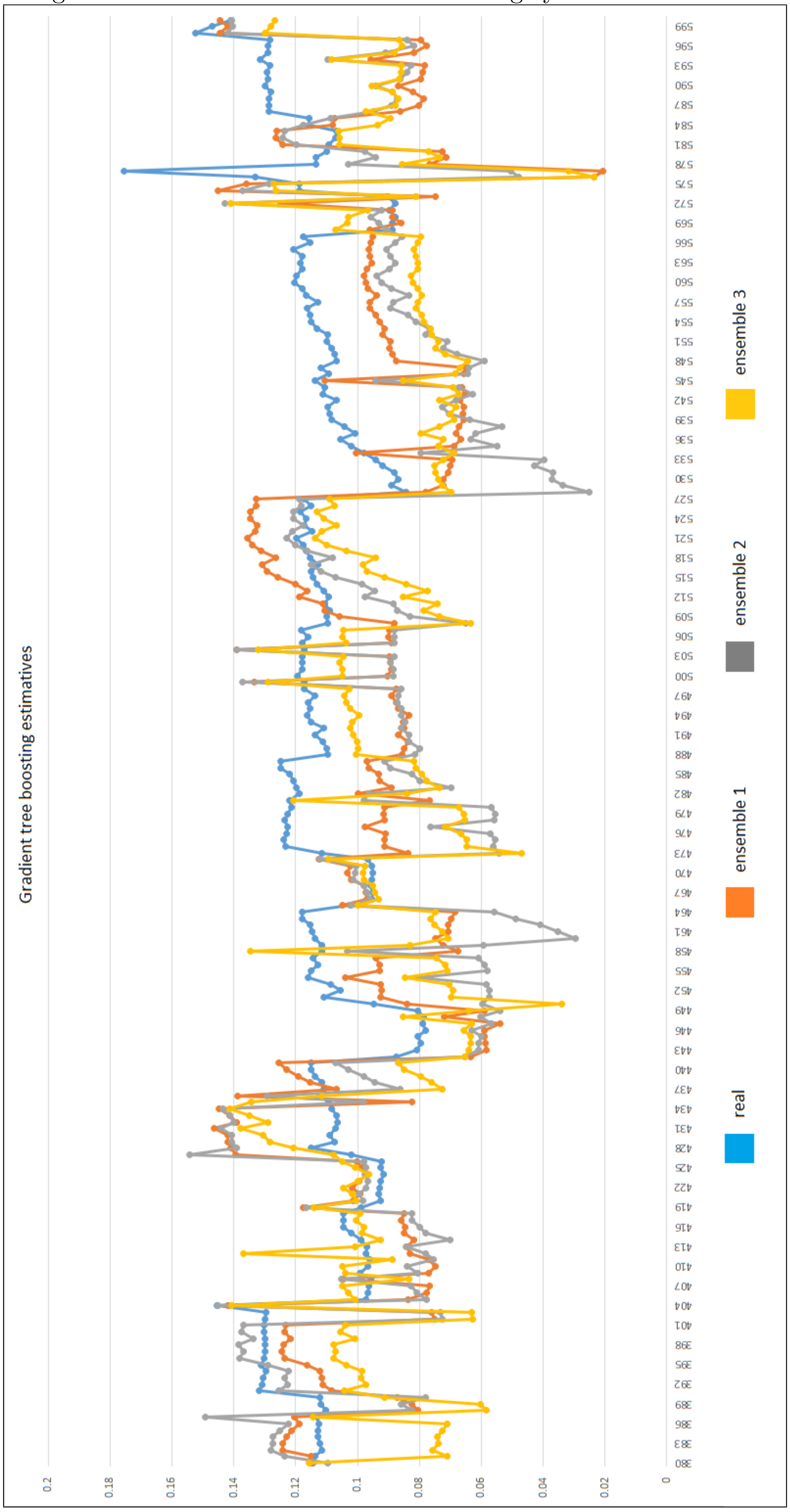
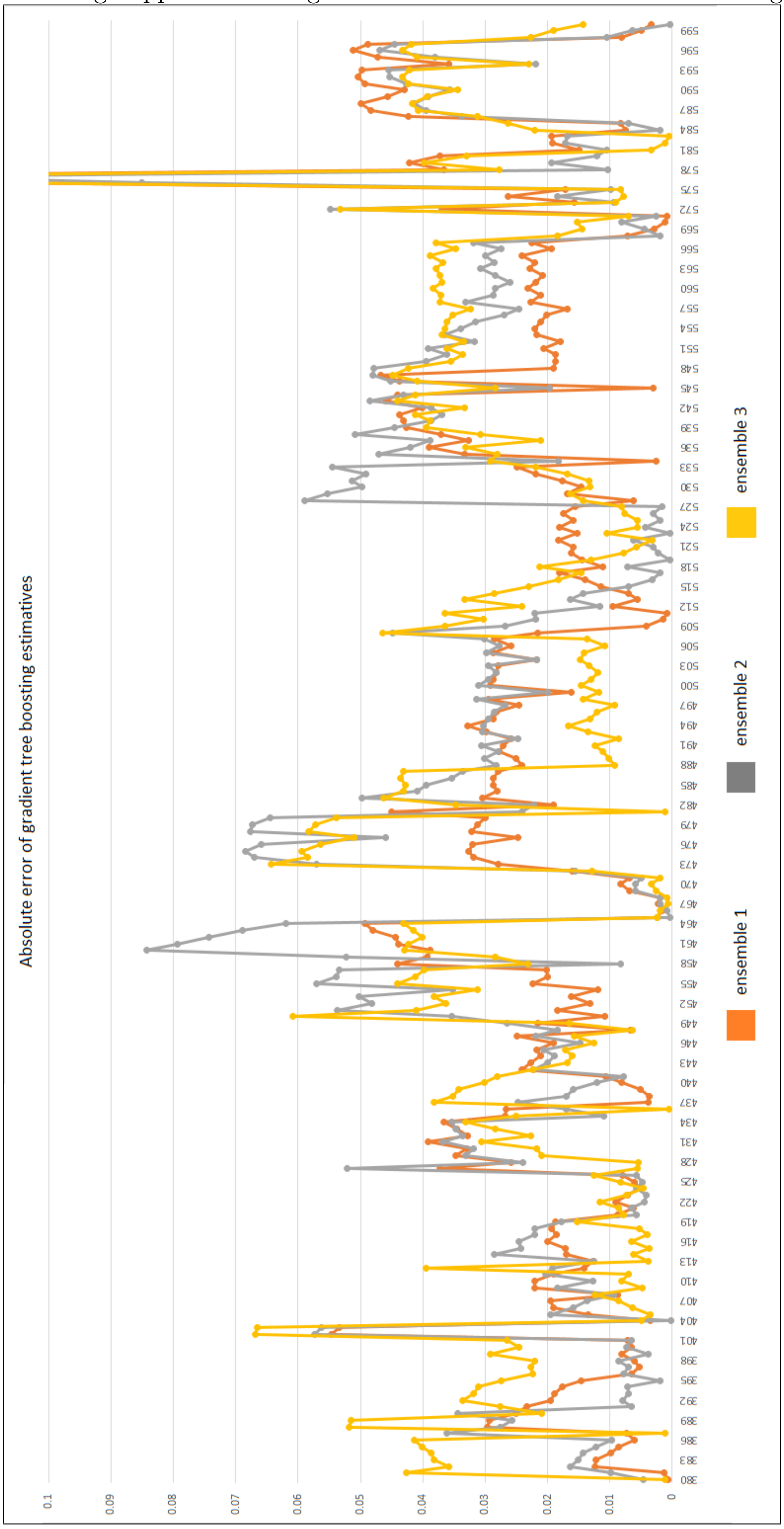


Figure 5.4 - Differences between estimated and real battery internal impedance values through application of xgBoost ensembles of models in the testing cycles



6 CONCLUSIONS

In this study, it was conducted an extensive benchmark analysis to enumerate and detail the various methods applied in the Li-ion battery SOC estimation. The described methods are categorized in: direct measurements, machine learning systems, and hybrid methods.

To estimate the Li-ion battery SOC, a well-known approach consists in the estimation of the Li-ion battery internal impedance related to the SOC parameter, according to Section 3.4. Some of the approaches which obtain Li-ion battery SOC parameter indirectly through the estimation of the Li-ion battery internal impedance are described in Section 3.2. A deeper investigation in the literature of these studies revealed two studies which perform tests in the same battery data set made available by National Aeronautics and Space Administration (NASA) Ames Prognostics Center of Excellence (SAHA; GOEBEL, 2007).

Herein, features which characterize the charge and discharge cycles of the battery data set were extracted from battery data set made available by National Aeronautics and Space Administration (NASA) Ames Prognostics Center of Excellence (SAHA; GOEBEL, 2007) and applied in the modeling process. In this stage, two broader applied regression techniques (MLP bagging and gradient tree boosting) received the training data sets and used in the deployment of two regression models.

Finally, the obtained regression models needed to be evaluated. Therefore, it was applied the $RMSE$ (Root Mean Square Error) performance metric, and the results were used to compare the two regression models.

In addition, the $RMSE$ error obtained in the estimation of the battery set impedance through MLP bagging and gradient tree boosting ensembles of models were compared with the $RMSE$ error of the benchmark model described in Section 1.2: Adaptive Neural Network ($AdNN$) (REZVANIZANIANI et al., 2011).

In order to compare the ensembles of regression models, the Student's t-test was applied, and the statistic difference between each other was defined (5.3.2).

According to Table 5.8 and considering the threshold of 0.05, the performance of the Gradient Tree Boosting ($xgBoost$) ensembles of models is higher when compared with the Multi-Layer Perceptron (MLP) bagging ensemble of models and the $AdNN$ benchmark model.

In addition to the adequate accuracy obtained by the gradient tree boosting ensemble of models, it also has the advantage of providing important insights regarding the data through the interpretation of the relationship between the extracted features. As detailed in Section 5.2.2, the importance of the features can be considered in the definition of the minimal features set that with a lower measurement effort makes possible the deployment of an ensemble of regression models with adequate accuracy. A future study can simplify the experimental laboratory setup by reducing the applied sensing. As described in Section 5.2.2, the mean Li-ion battery temperature during discharging cycles is the less relevant feature according to *xgBoost*) ensembles of models. In addition, the mean Li-ion battery temperature during charging cycles is the third less important feature. Thus, the temperature sensing can be removed, and the obtained ensembles of models can be tested to verify the impact of such modifications in their performance errors.

The estimation of the mean Li-ion battery internal impedance through Multi-Layer Perceptron (*MLP*) bagging and Gradient Tree Boosting (*xgBoost*) ensemble of models applying the feature extraction according to Section 4.3 had an adequate result. However, the estimation of the Li-ion battery internal impedance of battery sets operating under abnormal conditions (e.g. electric systems of hybrid cars) can reveal difficulties which were not found in the present study. Therefore, future studies could apply the proposed approach detailed herein in the estimation of Li-ion battery set internal impedance operating under various conditions.

In conclusion, the Multi-Layer Perceptron (*MLP*) and Gradient Tree Boosting (*xgBoost*) approaches described in this study were compared only with a machine learning model. Therefore, a future study could apply a direct measurement method in order to estimate the Li-ion battery internal impedance footprint of the same battery set and enrich the benchmark analysis detailed herein.

REFERENCES

ARMSTRONG, J. S. Illusions in regression analysis. **International Journal of Forecasting**, v. 3, p. 689, 2012. 5

ASLAM, J.; POPA, R.; RIVEST, R. On estimating the size and confidence of a statistical audit. **Proceedings of the Electronic Voting Technology Workshop**, v. 7, 2007. 7

BARBARISI, O.; VASCA, F.; GLIELMO, L. State of charge kalman filter estimator for automotive batteries. **Control Engineering Practice**, v. 14, p. 267–275, 2006. 26

BREIMAN, L. Bagging predictors. **Machine Learning**, v. 24, p. 123–140, 2003. 7

BREIMAN, L.; OLSHEN, R. A.; STONE, C. J. **Classification and regression trees**. USA: Wadsworth Brooks/Cole Advanced Books Software, 1984. 14, 18

CHANG, W. Y. State of charge estimation for lifepo4 battery using artificial neural network. **International Review of Electrical Engineering**, v. 7, p. 5874–5800, 2012. 25

CHIN; KEITH. Energy storage technologies for small satellite applications. **Proceedings of the IEEE**, v. 106, p. 419–428, 2018. 1

COLEMAN, M.; LEE, C. K.; ZHU, C.; HURLEY, W. G. State of-charge determination from emf voltage estimation: using impedance, terminal voltage, and current for lead-acid and lithium-ion batteries. **IEEE Transactions on Industrial Electronics**, v. 54, p. 2550–2557, 2007. 21

CYBENKO, G. Approximation by superpositions of a sigmoidal function mathematics of control. **Signals, and Systems**, v. 02, p. 303–314, 1989. 9

DALAL, M.; MA, J.; HE, D. Lithium-ion battery life prognostic health management system using particle filtering framework. **Proceedings of the Institute of Mechanical Engineering Part O J. Risk**, v. 225, 2011. 28

DATONG, L.; YUE, L.; YU, P.; XIYUAN, P.; MICHAEL, P. Lithium-ion battery remaining useful life estimation based on nonlinear ar model combined with degradation feature. **In: Annual Conference of Prognostics and Health Management Society, 2012. Proceedings...** 2012. 24

DUDLEY, G. J. Lithium-ion batteries for space. SVARTHOLM, N. (Ed.). **In: European Space Power Conference, 5., 1998. Proceedings...** 1998. p. 17. 1

EDDAHECH, A.; BRIAT, O.; VINASSA, J. M. Determination of lithium-ion battery state-of-health based on constant-voltage charge phase. **Journal of Power Sources**, v. 258, p. 218–227, 2014. 29

FADEM, B. **High-Yield Behavioral Science**. USA: Lippincott Williams Wilkins, 2008. 20

FARMANN, A.; WAAG, W.; MARONGIU, A.; SAUER, D. U. Critical review of on-board capacity estimation techniques for lithium-ion batteries in electric and hybrid electric vehicles. **Journal of Power Sources**, v. 281, p. 114–130, 2015. 28

FRIEDMAN, J. H. Additive logistic regression: a statistical view of boosting. **Annals of Statistics**, p. 337–374, 2000. 8

HANSEN, T.; WANG, C. J. Support vector based battery state of charge estimator. **Journal of Power Sources**, v. 141, p. 351–358, 2005. 26

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. H. **The elements of statistical learning: data mining, inference, and prediction**. USA: Springer, 2009. 24

HASTIE, T. T.; FRIEDMAN, J. H. R. Boosting and additive trees. : Springer, 2009. v. 02, p. 337–384. 17

HAYKIN, S. **Neural networks - A comprehensive foundation**. USA: Cambridge University Press, 1999. 10, 13, 24

HILL, C. A. Satellite battery technology — a tutorial and overview. **IEEE Aerospace and Electronic Systems Magazine**, v. 26, p. 38–43, 2011. 1

HYNDMAN, R. J.; KOEHLER, A. B. Another look at measures of forecast accuracy. **International Journal of Forecasting**, v. 04, p. 679–688, 2006. 19

KANG, L. W.; ZHAO, X.; MA, J. A new neural network model for the state-of-charge estimation in the battery degradation process. **Applied Energy 121**, v. 121, p. 20–27, 2014. 25

KIM, J.; CHO, B. H. State-of-charge estimation and state-of-health prediction of a li-ion degraded battery based on an ekf combined with a per-unit system. **IEEE Transactions on Vehicular Technology**, v. 60, p. 4249–4260, 2011. 27

KRUSE, R. Multi-layer perceptrons. : London: Springer, 2013. p. 47–81. 9

KUNCHEVA, L.; WHITAKER, C. Measures of diversity in classifier ensembles. **Machine Learning**, v. 51, p. 181–207, 2003. 6

KéGL, B. The return of adaboost: multi-class hamming trees. **University of Paris-Sud**, 2013. 7

LECUN, Y.; BOTTOU, L.; ORR, G. B.; MULLER, K.-R. Efficient backprop. : Springer, 1998. 11

LI, R.; WU, J. F.; WANG, H. Y.; LI, G. C. Prediction of state of charge of lithium-ion rechargeable battery with electrochemical impedance spectroscopy theory. **In: IEEE Conference on Industrial Electronics and Applications, 2010. Proceedings...** 2010. p. 684–688. 22

LINDA, O.; WILLIAM, E. J.; HUFF, M. Intelligent neural network implementation for soci development of li/cfx batteries. **Proceedings of the 2nd International Symposium on Resilient Control Systems**, 2009. 25

LIU, D.; WANG, H.; PENG, Y.; XIE, W.; LIAO, H. Satellite lithium-ion battery remaining cycle life prediction with novel indirect health indicator extraction. **Energies**, v. 6, p. 3654–3668, 2013. 28, 29

LIU, J.; SAXENA, A.; GOEBEL, K.; SAHA, B.; WANG, W. An adaptive recurrent neural network for remaining useful life prediction of lithium-ion batteries. **In: Annual Conference of the Prognostics and Health Management Society, 2010. Proceedings...** 2010. 25, 26

LIU, J.; WANG, G. A multi-step predictor with a variable input pattern for system state forecasting. **Mechanical Systems and Signal Processing**, v. 23, p. 1586–1599, 2009. 5

LIU, J.; WANG, M.; YANG, Y. A data-model-fusion prognostic framework for dynamic system state forecasting. **Engineering Applications of Artificial Intelligence**, v. 25, p. 814–823, 2012. 27

LORENZO, T.; LEVI, W.; CURTIS, H.; GIOVANNI, P. Bayesian nonparametric cross-study validation of prediction methods. **The Annals of Applied Statistics**, v. 1, p. 402–428, 2015. 19

MANSOURI, S. S.; KARVELIS, P.; GEORGOULAS, G.; NIKOLAKOPOULOS, G. Remaining useful battery life prediction for uavs based on machine learning. **IFAC Congress**, v. 50, p. 4727–4732, 2017. 24

MOSALLAM, A.; MEDJAHER, K.; ZERHOUNI, N. Nonparametric time series modelling for industrial prognostics and health management. **International Journal of Advanced Manufacturing Technology**, p. 1685–1699, 2013. 23

ONDA, K.; OHSHIMA, T.; NAKAYAMA, M.; FUKUDA, K.; ARAKI, T. Thermal behavior of small lithium-ion battery during rapid charge and discharge cycles. **Journal of Power Sources**, v. 158, p. 535–542, 2006. 30

PANDIT, S. M.; WU, S.-M. **Time Series and System Analysis with Applications**. USA: John Wiley Sons, 1983. 23

PARVIZ, M.; MOIN, S. Boosting approach for score level fusion in multimodal biometrics based on auc maximization. **Journal of Information Hiding and Multimedia Signal Processing**, v. 2, p. 51–59, 2011. 28

POP, V.; BERGVELD, H. J.; NOTTEN, P. H. L.; VELD, J. H. G. Op het; REGTIEN, P. P. L. Accuracy analysis of the state-of-charge and remaining run-time determination for lithium-ion batteries. **Measurement**, v. 42, p. 1131–1138, 2009. 27

REZVANIZANIANI, S.; ZONGCHANG, L.; CHEN, Y.; LEE, J. Review and recent advances in battery health monitoring and prognostics technologies for electric vehicle (ev) safety and mobility. **Journal of Power Sources**, v. 256, p. 110–124, 2014. 1

REZVANIZANIANI, S. M.; LEE, S.; LEE, J. A comparative analysis of techniques for electric vehicle battery prognostics and health management (phm). **SAE Technical Papers**, 2011. 3, 26, 31, 43, 47, 48, 49, 55

ROBERT, C. **Machine learning, a probabilistic perspective**. USA: CRC Press, 2014. 5

RON, K. A study of cross-validation and bootstrap for accuracy estimation and model selection. **In: International Joint Conference on Artificial Intelligence, 14., 1995. Proceedings...** 1995. v. 12, p. 1137–1143. 18

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning internal representations by error propagation. : MIT Press, 1986. v. 01. 11

SAHA, B.; GOEBEL, K. **Battery Data Set**. 2007. NASA Ames Research Center, Moffett Field, CA. Available from: <`http://ti.arc.nasa.gov/project/prognostic-data-repository`>. 2, 3, 28, 31, 32, 33, 35, 49, 55

SAHA, B.; GOEBEL, K.; CHRISTOPHERSEN, J. Comparison of prognostic algorithms for estimating remaining useful life of batteries. **Journal of the Institute of Measurement and Control**, v. 31, p. 293–308, 2009. 1

SALKIND, A. J.; FENNIE, C.; SINGH, P.; ATWATER, T.; REISNER, D. E. Determination of state-of-charge and state-of-health of batteries by fuzzy logic methodology. **Journal of Power Sources**, v. 8, p. 293–300, 1999. 26

SATO; KAWAMURA. A new estimation method of state of charge using terminal voltage and internal resistance for lead acid battery. **In: Power Conversion Conference, 2002. Proceedings...** 2002. p. 565–570. 22

SATO, N. Thermal behaviour or analysis of lithium-ion batteries for electric and hybrid vehicles. **Journal of Power Sources**, v. 99, p. 70–77, 2001. 30

SCHAPIRE, R. E.; FREUND, Y. **Boosting: foundations and algorithms**. USA: MIT Press, 2012. 8

SCHWABACHER, M. A survey of data-driven prognostics. **In: AIAA Meeting Papers, 2005. Proceedings...** 2005. 23

SHEEL, S.; VARSHNEY, T.; VARSHNEY, R. Accelerated learning in mlp using adaptive learning rate with momentum coefficient. **IEEE International Conference on Industrial and Information Systems**, p. 307–310, 2007. 13

SINGH, P.; VINJAMURI, R.; WANG, X.; REISNER, D. Design and implementation of a fuzzy logic-based state-of-charge meter for li-ion batteries used in portable defibrillators. **Journal of Power Sources**, v. 162, p. 829–836, 2006. 26

WANG, J.; CAO, B.; CHEN, Q.; WANG, F. Combined state of charge estimator for electric vehicle battery pack. **Control Engineering Practice**, v. 15, p. 1569–1576, 2007. 27

WATRIN; BLUNIER; MIRAOUI. Review of adaptive systems for lithium batteries state-of-charge and state-of-health estimation. SVARTHOLM, N. (Ed.). **Proceedings of IEEE Transportation Electrification Conference and Expo**. 2012. p. 1–6. 2, 21

WEBSTER, G. **Human error blamed for Mars probe failure**. April 2007. Available from: <http://www.abc.net.au/science/articles/2007/04/16/1897999.htm>. 1

WEIGERT, T.; TIAN, Q.; LIAN, K. State-of-charge prediction of batteries and battery-supercapacitor hybrids using artificial neural networks. **Journal of Power Sources**, v. 196, p. 4061–4066, 2011. 25

XU, L.; WANG, J. P.; CHEN, Q. S. Kalman filtering state of charge estimation for battery management system based on a stochastic fuzzy neural network battery model. **Energy Conversion and Management**, v. 53, p. 33–39, 2012. 26

YATSUI, M. W.; BAI, H. Kalman filter based state-of-charge estimation for lithium-ion batteries in hybrid electric vehicles using pulse charging. **In: IEEE Vehicle Power and Propulsion Conference, 7., 2011. Proceedings... IEEE**. 2011. p. 1–5. 26

ZHANG, J.; LEE, J. A review on prognostics and health monitoring of li-ion battery. **Journal of Power Sources**, v. 196, p. 6007–6014, 2011. 1, 21, 28, 35, 37