INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Departamento de Engenharia Eletrónica e de Telecomunicações e Computadores

Electric Vehicle X Driving Range Prediction – EV X DRP

David Alexandre Sousa Gomes Albuquerque

Licenciado

Projecto Final para obtenção do Grau de Mestre em Engenharia Informática e de Computadores

Orientadores : Doutor David Pereira Coutinho
Doutor Artur Jorge Ferreira

Júri:

Presidente: Doutor Tiago Miguel Braga Da Silva Dias

Vogais: Doutor Gonçalo Nuno de Oliveira Duarte
Doutor David Pereira Coutinho

December, 2022

Acknowledgments

First of all, I would like to thank my advisors Doctor David Pereira Coutinho and Doctor Artur Jorge Ferreira for their time and effort put into this project. I must also express my gratitude towards Instituto Politécnico de Lisboa and municipality of Oeiras for granting me a scholarship in which allowed me to further pursue my studies. I am also grateful to my classmates Miguel Luís, Nuno Gomes their teamwork and incentive. Finally, I would like to thank Rita Marques and my family for their motivation and support.

Abstract

The electric vehicle use as a reliable and eco-friendly means of transportation has increased rapidly over the past few years. When choosing an electric vehicle, its driving range capacity is a decisive factor to be taken into account as it minimizes driver's anxiety while driving.

An electric vehicle driving range depends on multiple factors that must be taken into account when attempting its prediction. Machine learning has become a widely used approach for highly complex problems, in which eRange prediction, being one of them, provides benefits such as becoming more accurate, the more the user drives his vehicle.

This thesis compares, through standard metrics, implementations of machine learning based regression models (Linear regression and Ensemble Stacked Generalization) when training with publicly available datasets.

The results of this work show the effects of different training sample sizes on machine learning model's accuracy and training time, presenting more favorable results for the Linear Regression algorithm, as the algorithm was more resistant to overfitting for commonly trained data. The results can be replicated with the implemented Python application, allowing for future testing and study of the topic.

Keywords: electric vehicle; range prediction; machine learning; energy consumption; dataset preprocessing

Resumo

A utilização veículos elétricos como um meio de transporte confiável e ecológico tem vindo a aumentar nos últimos anos. Ao escolher um veículo elétrico, o range elétrico de condução é um fator decisivo a ser levado em consideração, pois minimiza a ansiedade do utilizador durante a condução.

A autonomia de um veículo elétrico depende de vários fatores que devem ser considerados ao estimar a sua previsão. A aprendizagem automática tem sido uma abordagem amplamente utilizada para problemas altamente complexos, dos quais a previsão da autonomia do veículo, é beneficial para o consumidor ao tornar-se mais preciso quanto mais o veículo é utilizado.

Esta tese compara, através de métricas padrão e validação cruzada, implementações de modelos de regressão aprendizagem automática (*Linear Regression* e *Ensemble Stacked Generalization*) ao treinar com conjuntos de dados disponíveis publicamente.

Os resultados desta tese demonstram as alterações da qualidade de previsão e de tempo de treino que os modelos de aprendizagem automática sofrem quando são usadas configurações dos dados diferentes e demonstrando resultados mais favoráveis para o algoritmo de *Linear Regression*, pois este demonstra melhor resistência a sobreajustar aos dados mais comuns presentes no conjunto de treino. Utilizando a aplicação desenvolvida em Python, é possível a replicação resultados, promovendo estudos futuros no tema.

Palavras-chave: veículo eletrico; estimação de distância; machine learning; consumo energético; pré-processamento de datasets

Contents

List of Figures

List of Tables

Glossary

Artificial intelligence A general term used to describe the field of study of intelligent agents, which refers to any system that perceives its environment and takes actions that maximize its chance of achieving its goals .

Average energy consumption The average consumption of energy of an electric vehicle, this value is typically provided by manufacturer .

Big data A field that handles with large datasets that are too big and complex for traditional data processing .

Bushy tree A decision tree that has high variance results from high attribute splitting on with many values .

Dataset A structure containing data for a model.

Decision tree A supervised machine learning method for regression and classification, splitting the data into multiple branches .

Electric range Electric range, the maximum driving range of an electric vehicle using only power from its on-board battery pack to traverse a given driving cycle .

Electric vehicle A vehicle that depends on electric motors for its movement .

Ensemble A general meta approach to machine learning that seeks better predictive performance by combining the predictions from multiple models .

Ensemble stacked generalization An ensemble technique which combines multiple machine learning models's (base models) predictions through the usage of a meta model. The meta model is trained with base models' predictions from out sampled data as inputs and real values as outputs .

Extreme gradient boosting Also known as XGBoost is an open-source library that provides an efficient and effective implementation of the gradient boosting algorithm .

Full battery distance The maximum distance an electric vehicle can travel when its battery if fully charged, this value can be infered from the average energy consumption and the full battery energy which are typically provided by manufacturer .

Full battery energy The maximum charge an electric vehicle battery can store, this value is typically provided by manufacturer .

Gradient boosted decision tree The use of multiple decision (regression or classification) tree models in which, each model predicts the error from the previous model .

Gradient boosted regression tree The use of multiple regression tree models in which, each model predicts the error from the previous model .

Internet of things Describes physical objects (or groups of such objects) with sensors, processing ability, software, and other technologies that connect and exchange data with other devices and systems over a network .

K-means clustering An unsupervised machine learning algorithm which identifies k number of centroids, and then allocates every data point to the nearest cluster (collection of data points aggregated together because of certain similarities), while keeping the centroids as small as possible .

K-nearest neighbor A machine learning algorithm which averages training results from K number of neighboring features. The K parameter allows for configuring the model bias and variance ratio .

Knowledge extraction	Creation of knowledge from structured (relational databases, XML) and unstructured (text, documents, images) sources in a machine-readable and machine-interpretable format, representing knowledge in a manner that facilitates inferencing .
Light gradient boosted machine	LightGBM is a gradient boosting framework that uses tree based learning algorithms .
Linear regression	A statistical technique that is used to predict the outcome of a variable based on the value of one variable by finding a linear relation between the input variable and the output variable .
Machine learning	A branch of AI focused on learning from data.
Multiple linear regression	A statistical technique that is used to predict the outcome of a variable based on the value of more than one variables by finding a linear relation between the input variables and the output variables .
Neural network	A machine learning model that attempts to imitate the human brain to determine a solution for a given problem .
Python	A high level programming language.
Random forest	A supervised learning algorithm that uses ensemble learning method with decision trees that randomly selects features for the construction of the decision trees .
Regression tree	Decision tree that is used for the task of regression which can be used to predict continuous valued outputs instead of discrete outputs .
Reinforcement learning	An area of machine learning focused on managing intelligent agents that take actions with the intent of maximizing a reward .

Self-organizing map A type of artificial neural network that is trained using unsupervised learning to produce a low-dimensional, discretized representation of the input space of the training samples, called a map, and is therefore a method to do dimensionality reduction, applying competitive learning on each node, thus making each node compete for representation .

State of charge The the percentage of remaining battery charge, relative to the full battery energy .

Time series A series of data points indexed by time.

Acronyms

AEC	Average energy consumption.
DT	Decision tree.
EFB	Exclusive Feature Bundling.
ERange	Electric range.
EV	Electric vehicle.
EVA	Electric vehicles in action.
FBD	Full battery distance.
FBE	Full battery energy.
GBDT	Gradient boosted decision tree.
GBRT	Gradient boosted regression tree.
GOSS	Gradient-based One-Side Sampling.
JARI	Japan Automobile Research Institute.
LightGBM	Light gradient boosted machine.
MAE	Mean absolute error.
MAPE	Mean absolute percentage error.
MLR	Multiple linear regression.
MSE	Mean squared error.
MTT	Minimum trip time.

NDANEV National Big Data Alliance of New Energy Vehicles.
NN Neural network.

R2 Coefficient of determination.
RF Random forest.
RMSE Root mean squared error.
RT Regression tree.

SOC State of charge.
SOM Self-organizing map.

XGBoost Extreme gradient boosting.

1

Introduction

On today's day and age, the global concern on climate change has been a major focus on recent international agreements, such as the Paris Agreement (*Paris Agreement* 2015), incentivizing many car manufacturers to introduce electric vehicles (EVs) as the eco-friendly solution for sustainable transport for the future.

EVs have grown popularity in recent years and as a result, car manufacturers have increased competitiveness on vehicle's performance (Figenbaum *et al.*, 2015), namely the driving range capacity, as it is a decisive factor for consumers (Egbue and Long, 2012).

The EV's autonomy, designated here as electric range (eRange), allows consumers to know an estimate of the remaining driving distance for the existing EV battery power, easing driver's anxiety for the duration of a trip to a charging station (Smuts *et al.*, 2017; Song and Hu, 2021).

The eRange can be estimated through many driving data parameters, such as vehicle design, driver's behavior, weather, road inclination and state of charge (SOC) estimation. Its accuracy allows consumers to rely on its vehicle for longer travel time and efficient charging plans. However, eRange estimation is a complex problem with multiple influencing factors (Varga *et al.*, 2019) as demonstrated on Figure 1.1, fueling previous studies in the past to provide a solution for this challenge.

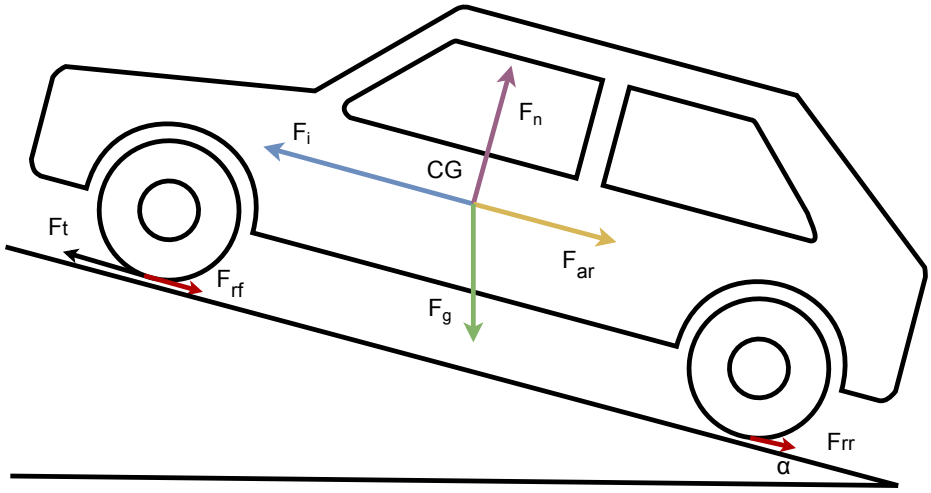

Figure 1.1: Main influencing forces on a moving vehicle. (F_i , inertial force; F_t , tractive force; F_g , gravitational force; F_{rr} , rear rolling resistance force; F_{fr} , front rolling resistance force; F_{ar} , aerodynamic (air) drag; F_n, normal force; CG, center Figure; α, the road slope)

The rise in popularity of machine learning (Amershi *et al.*, 2019) has demonstrated its effectiveness in the past with a variety of fields such as big data (Condie *et al.*, 2013; Zhou *et al.*, 2017), pattern recognition analysis and data mining (Bose and Mahapatra, 2001). This is due to its nature of learning from previous data to gradually achieve better results making it a widely recognized tool for complex problems (Mitchell, 2006).

As a result, previous works have used artificial intelligence, namely machine learning as a means for solving the eRange estimation problem, using different implementations through supervised, unsupervised and reinforcement learning (as it will detail on Section 2.3), making it a more accurate solution for its prediction.

1.1 Aim and Objective

This thesis focuses on providing a supervised machine learning eRange prediction model for any EV that can communicate with the application, given manufacturer vehicle information and externally monitored driving parameters. As future implementation, the developed Python application, could then be installed on an Internet of things device (such as a RaspberryPi). This would provide eRange estimations as

the vehicle sent the information in real-time, improving the prediction the further the vehicle drove.

An existing project named Classic eMini Project (Coutinho, 2021a) already provides some real-time battery and speed information required for eRange prediction algorithms to estimate its value.

1.2 Scope and Limitations

As this project will use supervised machine learning for the prediction of the eRange, datasets are required for training the model. As we do not possess such datasets gathered by ourselves, we depend on externally provided datasets.

The datasets must therefore also contain at least meaningful features in a time series format for the machine learning eRange prediction model training. These features include the state of charge (SOC), instantaneous energy consumption (IEC), speed, the timestamps and eRange. In the event of such features not being present on the training dataset as raw data, these must be inferred on a preprocessing phase for the dataset in question.

In the event that the eRange dataset feature is not available, existing eRange prediction algorithms such as Coutinho, 2021b could be used for estimating the eRange feature for the machine learning training target.

If so, it would require some additional information such as the full battery energy Full battery energy (FBE) and the Average energy consumption (AEC) supplied by the vehicle manufacturer for its algorithm.

This additional information is not present for the Classic eMini Project given that the vehicle is a transformed internal combustion engine vehicle (ICEV) to a battery electric vehicle (BEV), and no sufficient drive testing was conducted for its estimation.

As the project is designed to be later incorporated into resource limited hardware for the vehicle communication, an eRange machine learning prediction model could be favored according to its performance and thus hardware specifications must be taken into account for algorithm fine-tuning.

1.3 Organization of the Thesis

The remainder of this document is structured as follows. Chapter 2 refers the state of the art on existing eRange estimation solutions and their reliance on available datasets while mentioning machine learning and its usage on existing eRange estimation solutions. Chapter 3 will focus on detailing the resulting application, implemented algorithms, algorithm training, algorithm execution and algorithm evaluation. On Chapter 4 it is presented the project's results from algorithm comparison. Chapter 5 ends the document with a retrospective on the project development and its accomplishments regarding its main goals.

2

State of the Art

Nowadays, EVs have motivated multiple studies concerning related problems in this field, such as statistical measurement of charging (Brighente *et al.*, 2021), regenerative braking (Yoong *et al.*, 2010), charging topologies (Yilmaz and Krein, 2013) and eRange prediction (Varga *et al.*, 2019).

The eRange prediction is an important EV feature to present to consumers as it reduces driver's anxiety while driving. This has been previously studied before, prompting multiple ways to tackle the problem.

This chapter will introduce basic concepts (Section 2.1) and detail the literature and resources for the eRange prediction problem, machine learning concepts, the availability of public datasets as model training data (Section 2.2), eRange estimation solutions without machine learning (Section 2.3) and finally machine learning and its applicability in eRange prediction solutions (Section 2.3).

2.1 Concepts and Terminology

Most private consumer road transportation vehicles consist of internal combustion engine vehicles (ICEVs) and electric vehicles (EVs), the latter including battery electric vehicles (BEVs) and hybrid electric vehicles (HEVs).

ICEVs, are powered by fuels stored in a fuel deposit which must be transformed into power inside the vehicle. Although its method may vary, the fuel to power transformation has an efficiency of 40% to 60% due to part of its energy being wasted into byproducts such as heat.

BEVs on the other hand, store all of their energy inside lithium-ion rechargeable electric vehicle batteries (EVBs). These batteries' capacity are generally measured in kilowatt-hour - a composite unit of energy equal to one kilowatt (kW) sustained for one hour. When a battery is discharging its energy (in kWh), it does so at a certain power (kW), that is, the rate of energy transfer (kJ/s). The total energy *Etotal* discharged or charged by a battery during a certain time t_0 may be calculated by

$$E_{total} = \int_{t=0}^{t_0} P(t)\, dt\, [kWh]\,, \tag{2.1}$$

where P (t) is the instantaneous power at time t, or the rate of energy transfer at time t, expressed in kW. Regarding the vehicle energy consumption it is commonly expressed as kWh/100km.

The driving range estimate primarily relies on the amount of energy stored in the electric vehicle battery, which is most commonly represented by batteries state of charge (SoC). By definition the state of charge is the level of charge of an electric battery relative to its capacity. The units of SoC are percentage points where 0 and 100.

The direct measurement method, which is a simple current integration method, also known as "Coulomb Counting" (CoulCount), developed by Schmitt *et al.*, 2004. This method depends on a current measure obtained, for example, by using a DC current shunt resistor, that is connected between the battery pack and the motor controller.

According to the CoulCount method, to determine the delivered (or charged) energy $E_d(t_0)$ during a time interval t_0 , one should integrate during that time the measured battery current. More precisely, measuring the battery current during a time interval t0 and multiplying the measured battery current $I_b(t)$ by the battery voltage V_b, the result is the delivered (or charged) energy $E_d(t_0)$ during that time interval. Now, given that $P(t) = V_b(t) \times I_b(t)$, then using equation 2.1 we obtain the delivered (or charged) energy

$$E_{total} = \int_{t=0}^{t_0} V_b(t) \times I_b(t)\, dt \, [kWh] \,, \tag{2.2}$$

which is equivalent to apply equation 2.1 while using the measured battery current $I_b(t)$ with the DC current shunt resistor.

So, the battery level of charge or remain battery energy (RBE) is estimated by

$$RBE = FBE - E_d(t_0) \, [kWh] \,, \tag{2.3}$$

where FBE stands for full battery energy and is the useable vehicle battery capacity provided by the vehicle manufacturer. Finally, the state of charge (SoC) expression is derived

$$SoC = \frac{RBE}{FBE} \times 100 \, [\%] \,, \tag{2.4}$$

where the SoC values ranges from 0 to 100. Because this method suffers from long-term drift and lack of a reference point, the SoC must be re-calibrated on a regular basis. Therefore, every time the charger determines that the battery is fully charged the SoC should be reset to 100.

Some variations or enhanced Coulomb counting methods have been successfully applied for the SoC estimation of lithium-ion batteries, while conventional battery SoC estimation methods either involve sophisticated models or consume considerable computational resources (Xie *et al.*, 2018).

2.1.1 Basic Driving Range Prediction

Other studies have focused on delivering higher eRange estimation accuracy, making use of more complex models. The *Classic EV X Project Driving Range Prediction* (Coutinho, 2021b) project proposes two eRange estimation algorithms: the *"Basic"* (Figure 2.1) and the *"History-based"* (Figure 2.2).

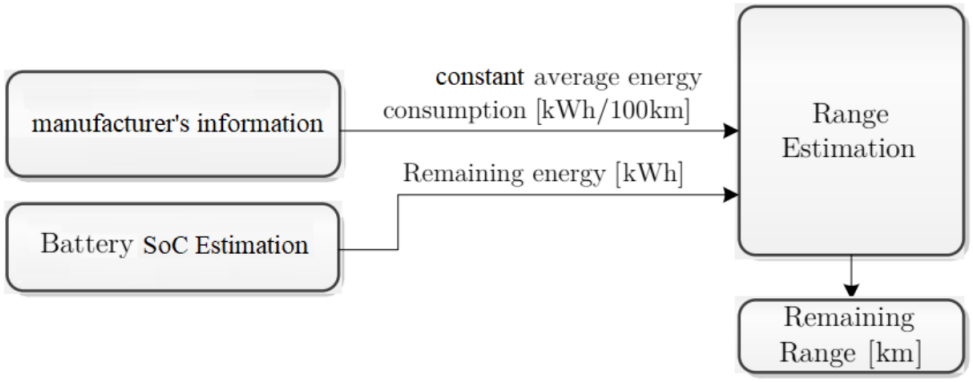

Figure 2.1: *"Basic"* range estimation system (modified from Enthaler and Gauterin, 2015).

The *"Basic"* algorithm calculates eRange through the combination of the EV model's constant information provided by the vehicle's manufacturer such as: the maximum charge an electric vehicle battery can store, known as FBE; and the manufacturer's provided constant average consumption of energy of an electric vehicle (AEC_C), which depends on the air-conditioner (AcS) and type of trip (highway or city driving). It also requires the state of battery charge (SOC) at the time of the eRange estimation as well. So, the eRange is given by

$$eRange(AcS, AEC) = \frac{FBE}{AEC_C(AcS)} \times SOC\ [km]\ . \tag{2.5}$$

When proving a solution to the eRange prediction problem, valid EV driving data in the form of a dataset is required to learn and evaluate the proposed model, and compare it to existing alternatives, making it indispensable in determining the effectiveness of the chosen solution.

2.1.2 Machine Learning Concepts

Machine learning is a field in which the focus is on data learning automation without human intervention. Generally, finding the best perceived learning technique and then learning the model from the data. Choosing a learning technique, is dependent on the machine learning type which could be either be unsupervised, supervised or reinforcement learning.

Reinforcement learning is an area of machine learning focused on managing intelligent agents that take actions with the intent of maximizing a reward. This area typically relies on reward algorithms to calculate each decision's reward. As the algorithms attempt to maximize reward, the better gradually achieved solution is.

Unsupervised learning is typically used in situations where the real value cannot be observed for prediction comparison, or when the objective is to study the relation between the data (e.g.: clustering data).

For supervised learning, the expected output data is already known, and the algorithms are evaluated on how close to the observed real value the prediction was. Problems such as classification and regression fall on this category.

Classification algorithms are used when a specific set of possible output scalar values or labels are strictly defined, e.g.: possible colors; predicting an age, etc.

Regression algorithms are only applied to real value estimation, e.g.: predicting medication dosage in relation to patient weight.

Due to machine learning models requiring data to be trained, their effectiveness derives from the quality (and relevance) of the training data. In some scenarios, the model can perform very well for training data, but perform poorly to test data, this is known as having a high predictive variance, and is caused by overfitting the model to the training data. To solve this problem the model must induce bias to the training data, which in turn reduces the predictive variance. In some algorithms, finding controlling the balance between bias and variance can be done on what is known as a regularization term.

2.2 Public Domain Datasets

In order to learn machine learning prediction models, existing datasets are available for use in supplying the needed training features for an eRange prediction model. Some of these being instant such: as SOC; speed; acceleration; battery energy consumption and road inclination (or elevation). While others being constant for the duration of the trip: vehicle information as battery capacity, AEC (Average energy consumption), max eRange, weight; trip information such as commute type city or highway; total energy consumption and total distance.

Datasets like *VED* dataset (Oh *et al.*, 2019) that although providing 54 different EV driving trip data for estimation, lack trip and vehicle information as well as sufficient EV model variety, with only three distinct EVs present on its dataset, all from the same model 2013 *Nissan leaf*.

The *Charge Car* project of the CREATE Lab at Carnegie Mellon University (*ChargeCar Database* n.d.) publicly supplies crowd-sourced data that has served previous eRange prediction models before (Zheng *et al.*, 2016). This dataset has a high vehicle diversity due to the open nature of the platform, allowing any user being able to upload combustion engine based vehicle information as well the location data, speed, weather as well other parameters, battery information could be supplied through CREATE RAV4-recorder box (*ChargeCar's CREATE RAV4-recorder box* n.d.). The EV consumption information is inferred through the platform using its simulation algorithm. The location, trip and vehicle information are then used to determine the simulated EV consumption for each trip. As of the time of writing, a total of 373 unique trips are openly available as a dataset option.

Another dataset solution is the *Emobpy* Python tool (Gaete-Morales *et al.*, 2021) that focuses on EV trip and charge data generation through empirical mobility statistics and customizable assumptions. Although this approach has by definition an infinite supply of EV trips as well as proper vehicle information, this dataset is lacking in some trip parameters such as speed, elevation, trip and commute type.

A dataset collected through probe data from nearly 500 BEVs by the Japan Automobile Research Institute (JARI) between February 2011 to January 2013 by the JARI (*Project Consigning Technology Development for Rational Use of Energy (Innovative Manufacturing Process Technology Development)* n.d.) allegedly consists of the following features: time; location; vehicle state (driving, normal charging or fast charging); speed; air-conditioner and heater state; SOC. Although useful and featured in some papers (Sun *et al.*, 2015; Sun *et al.*, 2016; Liu *et al.*, 2017; Liu *et al.*, 2018), this study was unable

to acquire this dataset from (*JARI research database* n.d.) perhaps due to the language barrier.

Previous studies in of eRange prediction (De Cauwer *et al.*, 2017) have depended on *EVteclab*'s Electric vehicles in action (EVA) platform, a Flemish Living Labs project (*EVteclab's eva platform* n.d.). The platform supplies a dataset containing monitoring data of 30 different model *Ford Connect* EVs for the duration of a year. This dataset although, supplying a few useful parameters like timestamp, latitude, longitude and vehicle speed, was inaccessible at the time this of writing.

The cloud based EV dataset supplied by the National Big Data Alliance of New Energy Vehicles (NDANEV) (*National Big Data Alliance of New Energy Vehicles* n.d.) has been used in a similar eRange prediction approaches (Zhao *et al.*, 2020). The data was collected from Controller Area Network (CAN) of five different EVs of an undisclosed model through with T-BOX, later uploading it to NDANEV. The dataset distinguishes from the others by including battery cell temperature information, a vital feature for includes many valuable parameters, including cell temperature information being one of them which has been previously used in the measure of battery cell inconsistency.

As some datasets do not supply the vehicle information directly, the ev-database (*Electric Vehicle Database* n.d.) website supplies an easily accessible database for existing EVs, displaying AEC, total eRange and usable battery energy. Through this information, some otherwise information lacking datasets can be used in eRange prediction models.

Table 2.1 describes most features present in currently accessed public datasets, the * character indicates the feature is not always present.

Table 2.1: Dataset features.

	VED dataset	Emobpy	Classic EV X Project	ChargeCar	NDANEV
Trips	507	Unlimited	1	373	2372
EV Models	1	102	1	?	1
EVs	3	N/A	1	?	5
Features	timestamp, speed, location, battery SOC, battery voltage, battery current, AC power, heater power, OAT	timestamp, distance, IEC, consumption, average power, state	timestamp, IEC, RBE, speed	timestamp, elevation, planar distance, adjusted distance, speed, acceleration, model power, actual power*, current*, voltage*	timestamp, speed, total voltage, total current, battery SOC, temp. range, motor voltage, motor current, mileage

2.3 EV Range Prediction Techniques

ERange has been an interesting topic in recent years, in part due to increase in EV usage by the consumers as they become more efficient. As previously discussed, its prediction complexity is in part a due to the fact that there are multiple factors to take into account when measuring it, such as battery and road information, previous vehicle trips, weight. This has motivated researchers in finding solutions for the problem, and even tapping into machine learning for its prediction.

Existing work has demonstrated the use of eRange estimation on EVs, showing the need for different types of accuracy on eRange estimation depending on the SOC state, the proposed approach by Zhang *et al.*, 2012 minimizes the performance impact of minimum cost route searching from high accuracy eRange prediction.

The use of machine learning for a multitude of cases (Amershi *et al.*, 2019) in fields such as big data (Condie *et al.*, 2013; Zhou *et al.*, 2017) and data mining (Bose and Mahapatra, 2001) has proven its robustness on solving complex problems. As a result, some approaches for the eRange problem have already applied machine learning, most notably supervised learning to achieve the estimation.

The remainder of this chapter will take an overview on the eRange prediction algorithms, starting by the non machine learning adaptive *"History-based"* algorithm and then describing the remaining machine learning algorithms.

2.3.1 Adaptive History-based Algorithm

The *"History-based"* method is commonly used in range estimation for EVs (Enthaler and Gauterin, 2015) and is depicted in Figure 2.2.

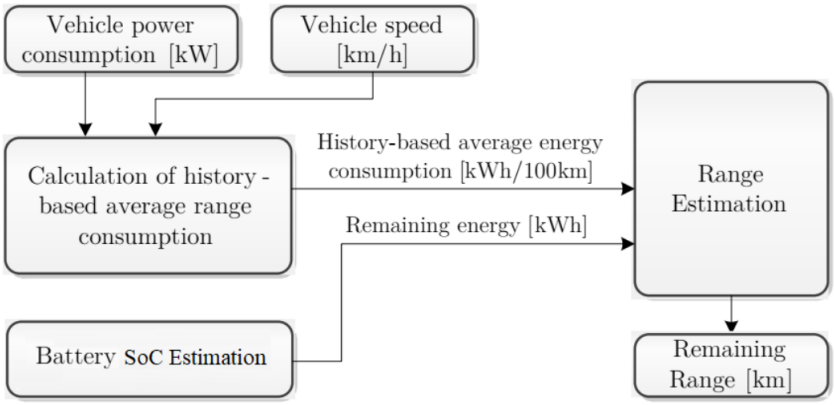

Figure 2.2: *"History-based"* range estimation system (modified from Enthaler and Gauterin, 2015).

An adaptive algorithm of this method was introduced in Coutinho, 2021b. It relies on parameters like the full battery energy (FBE), state of charge (SoC), where an instant energy consumption (IEC) is used to calculate every minute an adaptive value for AEC. So, the remaining eRange for the minute k is determined, based on Equation (2.5), as follows

$$eRange(k) = \left\lfloor \frac{FBE}{\sum_{i=0}^{N-1} w_i \times AEC_A(k-1)} \times SoC(k) \right\rfloor , \tag{2.6}$$

where N is the number of past minutes of an observation moving window and w_i are the predefined weights to the moving average computation of each minute's adaptive AEC (AEC_A). Exponentially less weights are assigned to less recent AEC values, so the weight values are as shown in the following expression,

$$w_i(k) \in \left\{ \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, ..., \frac{1}{2^N}, \frac{1}{2^N} \right\} . \tag{2.7}$$

The algorithm however, requires three additional constant parameters, which are also required: the delta power step (ΔS), the constant AEC (AEC_C), as provided by the vehicle's manufacturer and the minimum instance energy.

The ΔS represents the amount of power the AEC_A fluctuates for each k minute fluctuates. It is used to compute $AEC_A(k)$ according to the following expression

$$AEC_A(k) = \begin{cases} AEC_C, & k <= N \\ AEC_A(k-1) - \Delta S, & AEC_{ma}(k) < 0\,, \\ AEC_A(k-1) + \Delta S, & AEC_{ma}(k) > 0 \end{cases} \tag{2.8}$$

by adding or subtracting the pre-configured delta step (ΔS) to the previous AEC_A calculation of the previous k minute. Initially, AEC_A is equal to the pre-configured AEC_C, until it is possible to calculate the moving average with minimum number of N samples. In this equation, AEC_{ma} represents the moving average of the current k minute IEC values, where every non-zero instantaneous energy consumption (IEC) value is averaged for its calculation.

The minimum instance energy's role is to prevent the algorithm from performing an eRange calculation when the average IEC values for the current k minute are less than a predefined threshold value. This is done so that in the event the vehicle consumes negligible power, it would not cause an ΔS decrement or increment on the eRange prediction, thus preventing inaccurate eRange results.

The *"History-based"* algorithm is regarded as an improvement, yielding slightly more optimistic values than the *"basic"* algorithm. This is due to the algorithm's adaptation to the vehicle's current usage, and thus easing consumer's anxiety when a higher energy consumption affected the latter more proportionally.

As one this thesis' goals is the future integration with the Classic eMini Project project (Coutinho, 2021a) and given that there are no real energy data collections available for the eMini yet, the *"History-based"* algorithm was chosen as the training target for our machine learning models (as it will be detailed on Section 3).

2.3.2 Ensemble Stacked Generalization

The use of decision trees (DTs), random forests (RFs), and K-nearest neighbor in ensemble stacked generalization (ESG) approach, through the *JARI* dataset (in 2.2) (Ullah *et al.*, 2021) has proved its effectiveness in yielding more acceptable values in overfitting cases for proposed evaluation metrics.

Ensemble learning is a general meta approach to machine learning that seeks better predictive performance by combining the predictions from multiple models (base models, in this case DTs, RFs, and K-nearest neighbor). In this approach, stacked generalization is an ensemble method where a different model (meta model, in this case AdaBoost), learns how to best combine the predictions from multiple existing models, The meta model is trained with base models' predictions from out sampled data (not present on base models' training set) as inputs and real values as outputs, where in this case, Stratified K-Fold Cross Validation is used. The overview of this approach can be seen on the following Figure 2.3.

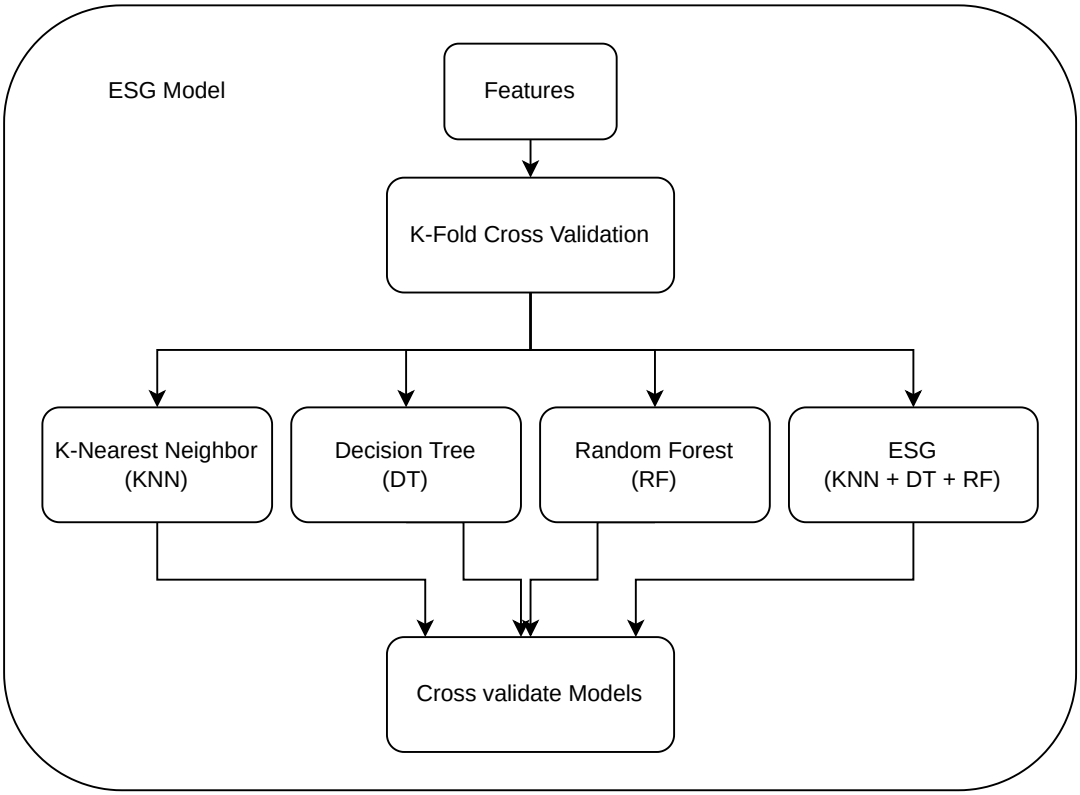

Figure 2.3: Ensemble Stacked Generalization model Ullah *et al.*, 2021.

Decision tree (DT) is a supervised learning data structure that provides a regression (as in this case, they are also known as RTs) or classification model that is used to predict outcomes from multiple variables. The data structure consists of multiple internal nodes which are used for splitting the data based on condition rules to maximize the information gain, and leaf nodes represent decisions for the prediction. A condition for the split is chosen by selecting a predictor candidate that has the lowest residual value from any given threshold value. An example of a DT is depicted on Figure 2.4, where is shown the association between Energy kcal/ day, hunger, wanting, restrained eating (rest.eat) and relative reinforcement of food (rrvf) from the *Box Lunch Study* Venkatasubramaniam *et al.*, 2017:

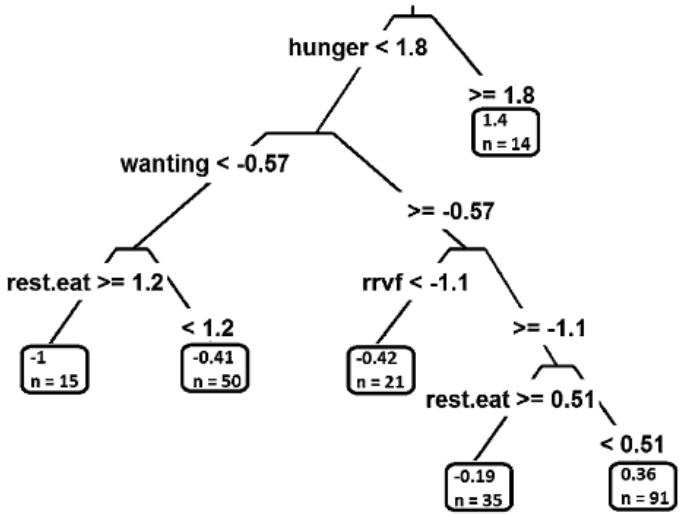

Figure 2.4: Regression Tree (RT) example showing the relation between Energy kcal/ day, hunger, wanting, restrained eating (rest.eat) and relative reinforcement of food (rrvf) from the *Box Lunch Study* (modified from Venkatasubramaniam *et al.*, 2017).

K-nearest neighbors (K-nearest neighbor) is a machine learning algorithm which averages training results from K number of neighboring features. The K parameter allows for configuring the model bias and variance, if K is small, bias gets lower and variance gets higher, the opposite effect is seen if K is higher, due to more neighbors for the averaging features.

Random forest (RF) is an ensemble method which constructs multiple DTs from the same dataset but with randomly selected entries for each tree (bootstrapped dataset) and randomly selecting features. The bootstrapped dataset allows for repeated samples and the process is exemplified on Figure 2.5.

Original Dataset

Feature 0	Feature 1	Expected
0	0	A
0	1	B
1	0	C
1	2	D

Bootstrapped Dataset

Feature 0	Feature 1	Expected
0	1	B
1	2	D
1	0	C
1	2	D

Figure 2.5: Dataset bootstrapping example.

After constructing the DTs, each tree will predict the outcome for the given input variables and provide a prediction which will be aggregated for the majority voting of common predictions (known as bagging). This algorithm reduces the DT limitations of being sensitive to training though randomly selected training entries as well as tree correlation with random feature selection. An example of the RF algorithm is depicted on Figure 2.6.

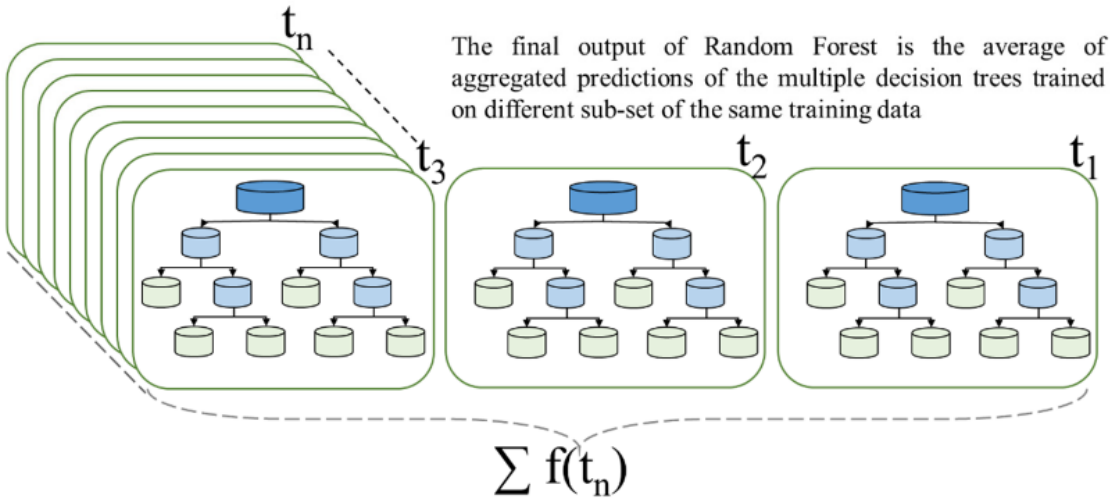

Figure 2.6: Random forest (RF) example (modified from Shah *et al.*, 2019).

K-Fold Cross Validation is an estimation algorithm that helps evaluate the error of a given prediction model. This algorithm is often used for model fitness comparison, as it prevents validation bias by dividing the training data into K subsamples. After which, every subsample will be selected for validation or training K times, one for each model. Selection bias is mitigated as all subsamples will be used for validation on their own K fold. For example, if K=3, then the dataset used for training and validation will be split into 3 subsamples, and 3 models will be trained and validated by each sample Figure 2.7.

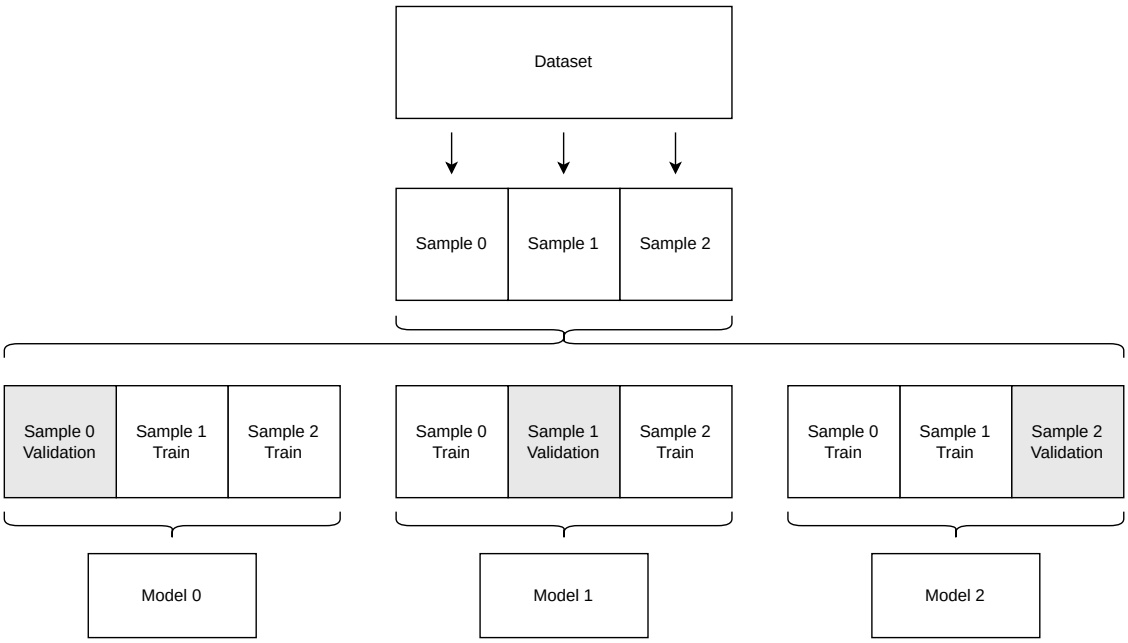

Figure 2.7: K-Fold Cross Validation example with K=3.

In this case, Stratified K-Fold Cross Validation is a variant of the K-Fold Cross Validation in which the average response data from each training subsample is approximately the same in all the partitions.

AdaBoost (or Adaptive boosting) is a meta algorithm that constructs forests of decision trees, consisting of only a root node and two leafs (decision stump tree). Each decision stump is placed on an order of prediction and have an α value which determines the amount of say a stump has on the final estimation. This value will be adjusted according to the decision stump tree's prediction mistakes. The dataset samples have a configured weighted probability value which is used for bootstrapping the dataset for the next stump training.

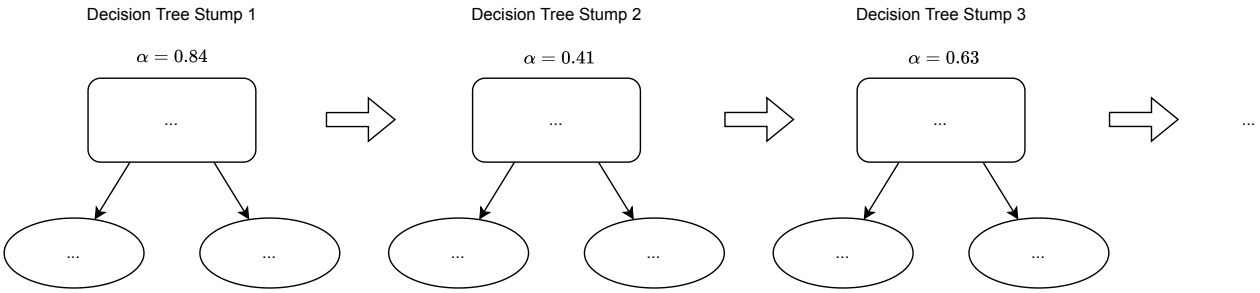

Figure 2.8: AdaBoost forest of Decicion tree stumps and their amount of say in the final estimation represented by the α value.

The probability increases with previous stump estimation errors, thus increasing the change the next stump to adapt to the previously problematic samples. The more predictive error the stump has, the less is the α value, favoring better predictive stumps that have a higher α value. The Figure 2.8 illustrates the decision stumps and their respective α value. The ensemble stacked generalization model uses the *SAMME.R* implementation from Zhu *et al.*, 2006 that naturally extends the original AdaBoost algorithm to the multi-class case without reducing it to multiple two-class problems.

2.3.3 XGBoost with LightGBM

Recent models using gradient boosted regression trees (GBRTs) have combined extreme gradient boosting (XGBoost) and light gradient boosted machine (LightGBM) to provide better predictive performance from these ensemble methods (Zhao *et al.*, 2020) with the NDANEV dataset (Section 2.2), the approach classified four different the driving patterns from three different parameters: speed; motor current; change rate of motor current, through K-means clustering algorithm and thus influencing result eRange due to their different energy consumption rates, (source code in *Machine Learning-Based Method for Remaining Range Prediction of Electric Vehicles - source* n.d.). The system diagram for the blended model is depicted on the Figure 2.9 bellow.

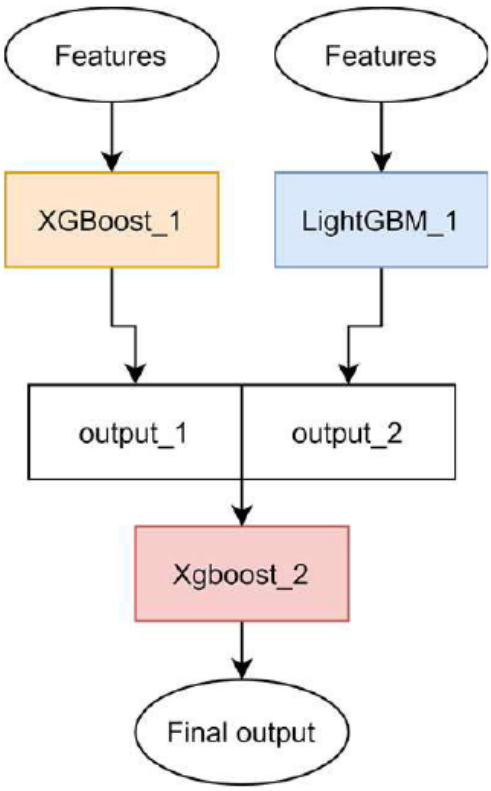

Figure 2.9: XGBoost + LightGBM blended model (from Zhao *et al.*, 2020).

Both XGBoost and LightGBM are gradient boosting algorithms, taking advantage of the output of multiple trees to aggregate a value. The XGBoost, algorithm uses an objective function that is defined in the paper's approach,

$$obj(\theta) = L(\theta) + \Omega(\theta) \,, \tag{2.9}$$

where θ denotes the parameters that the regression model has learned, $obj(\theta)$ is the goal of finding the minimum output value for the leaf that minimizes the equation, $L(\theta)$ is training loss term and $\Omega(\theta)$ is the regularization term. The training loss term is given by

$$L(\theta) = \sum_{i=1}^{n} [L(y_i, \hat{y}_i^{t-1}) + g_i f_t(x_i) \frac{1}{2} h_i f_t^2(x_i)] \,, \tag{2.10}$$

where y_i is the observed value and $\hat{y}_i^{t-1}$ represents the previous i^{th} prediction. g_i and h_i are the first and second derivatives of $L(y_i, \hat{y}_i^{t-1})$ to $\hat{y}_i^{(t-1)}$ respectively. The training process follows an additive manner defined by

$$\hat{y}_i^t = \hat{y}_i^{(t-1)} + f_t(x_i) \,, \tag{2.11}$$

in which, $\hat{y}_i^t$ denotes the prediction of the i^{th} instance at the t^{th} iteration. The tree $f_t(x)$ is expressed by

$$f_t(x) = \omega_q(x) \,, \tag{2.12}$$

where ${}_q(x)$ is a function that assigns each data point to a corresponding leaf and ω, the vector of scores on leaves.

Regularization typically helps preventing variance on test data by inducing bias to the training data. The regularization term $\Omega(\theta)$ of the objective function is defined by

$$\Omega(\theta) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} \omega_j^2 \,, \tag{2.13}$$

where γ is a user defined constant penalty that encourages tree pruning, T is the number of terminal nodes (leafs) in a tree, λ is user defined constant value that scales the output and ω_j is the output value from the tree.

LightGBM (Ke *et al.*, 2017) on the other hand, achieves similar results with a substantially faster training time. This is possible due to multiple optimizations made when comparing to the XGBoost on the algorithm, namely Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB).

In gradient boosted decision trees (GBDTs), sample weights are not natively available to determine data importance as in the AdaBoost algorithm. GOSS however, takes the advantage of the gradient-importance relation in GBDTs, where if an instance is associated with a small gradient, the training error for this instance is small and therefore, already well-trained. The algorithm improves on GBDTs as the histogram-based algorithm needs to retrieve feature bin values, for each data instance no matter the feature value is zero or not.

As high-dimensional data is usually very sparse, the feature space characteristic in which, many features are mutually exclusive (never taking nonzero values simultaneously), has let to the inception of the EFB algorithm. The algorithm uses this feature property to bundle exclusive features into a single feature, effectively reducing complexity from $O(\#data \times \#feature)$ to $O(\#data \times \#bundle)$, where $\#bundle << \#feature$.

Although XGBoost and LightGBM are similar gradient boosting algorithms, in Zhao *et al.*, 2020 the algorithms are combined to improve on the first models' predictions with a second layer XGBoost, further increasing the predictive performance of the blended model.

2.3.4 Hybrid Self-Organizing Maps with Regression Trees

The hybrid version of self-organizing map (SOM) (Kohonen, 1990) with RTs (Zheng *et al.*, 2016) has taken advantage of SOM neurons storage feature of nearing related neighbor information being kept closely together, performing RTs locally and avoiding bushy trees, keeping meaningful knowledge extraction on bushy trees.

A SOM is an unsupervised machine learning technique in which an artificial neural network is trained through competitive learning. An artificial neural network is typically represented by a cluster of neuron cells. Competitive learning makes the neurons compete with each other for representation of the input features.

When constructing a SOM, the neurons have a weight vector for each input. This weight is initialized randomly and will be adjusted towards input data as the algorithm progresses, reducing a configured distance metric.

The closest a neuron vector is to an input, the higher the chance it has to become the best matching unit (BMU). Once a best matching unit for a given input is chosen, the algorithm updates both its weight, and neighboring neurons' weights. As defined in Kohonen, 1990, updating a neuron's input weight is given by

$$m_i(t+1) = m_i(t) + h_{ci}(t)[x(t) - m_i(t)] \,, \tag{2.14}$$

where i is an input-neuron vector, m_i is the weight of neuron, t the step index, x the input vector and h_{ci} is defined by

$$h_{ci}(t) = \alpha(t) \,, \tag{2.15}$$

where $\alpha(t)$ is a suitable, monotonically decreasing sequence of scalar-valued gain coefficients, $0 < \alpha(t) < 1$. Figure 2.10 demonstrates the neighboring cells N_c.

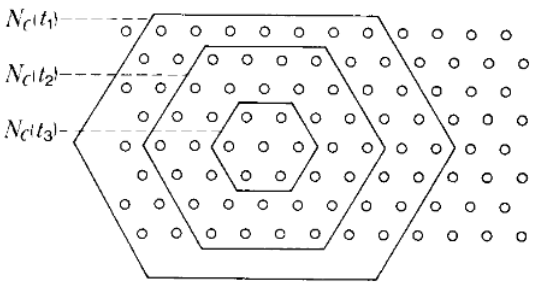

Figure 2.10: Example of topological neighborhood in Self-organizing maps
N_c - neighboring cell, t - time step, where $t_1 < t_2 < t_3$ (from Kohonen, 1990).

Once the algorithm finishes, neighboring neurons keep the closest inputs clustered together, in which the hybrid version of SOM with RTs (Zheng *et al.*, 2016) takes advantage of, creating RTs from these related neighbors. The paper referenced an undisclosed "EV Cloud Platform" which was used for obtaining the dataset, for training and testing. The hybrid algorithm diagram can be seen on Figure 2.11.

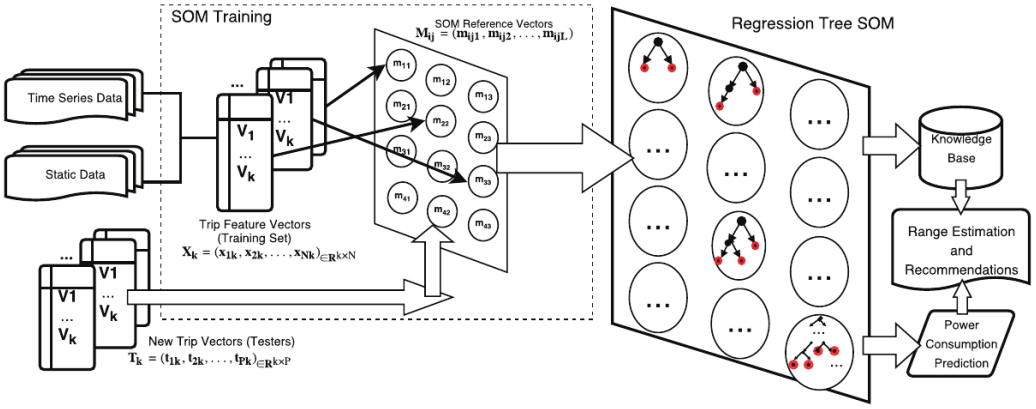

Figure 2.11: Self-organizing maps + Regression tree model (from Zheng *et al.*, 2016).

2.3.5 Growing Hierarchical Self-Organizing Maps

Approaches using unsupervised clustering of SOMs have been used for clustering big data into driving patterns, prior to range estimation through Growing Hierarchical Self-Organizing Map (GHSOM) (Lee and Wu, 2015). For the dataset construction, the approach used 787 records from EVs in Taiwan for over one year, as well as a 106Ah2S Li-ion battery module of a production EV, which was evaluated to model the aging trend and long-term performance of the EV battery pack. The system diagram can be seen bellow on Figure 2.12.

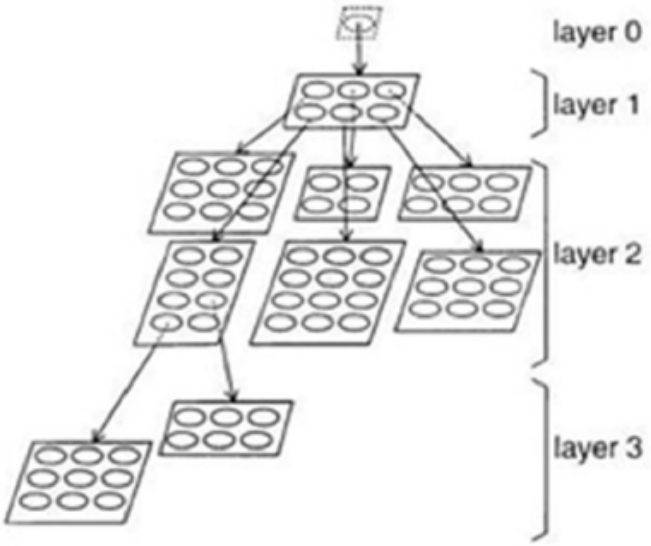

Figure 2.12: Growing hierarchical self-organizing maps model (from Lee and Wu, 2015).

The GHSOM (Rauber *et al.*, 2002) algorithm is an improvement of the Kohonen, 1990 algorithm. The algorithm takes into consideration the limitations of the SOMs. One of such limitations is the fixed network architecture in terms of the number and arrangement of neural processing elements, which has to be defined prior to training.

Another limitation is the fact that hierarchical relations between the input data are not mirrored in a straight-forward fashion. Such relations are rather shown within the same representation space and are thus hard to identify.

To overcome these limitations, the algorithm creates multiple SOMs in an hierarchical fashion. The algorithm begins by creating a SOM with a single neuron (layer 0) then, from that neuron, a new small SOM is created (layer 1).

For every neuron in the layer, the mean quantization error of the neuron, mqe_n is calculated by

$$mqe_n = \frac{1}{N} \sum_{k \in X_n} ||w^k - S_n|| \,, \tag{2.16}$$

where X_n is the set of training vectors that label to neuron n, w^k is the k^{th} training vector, N is the number of training vectors and S_n is the synaptic weight vector of neuron n.

The mqe_n of every neuron in the SOM layer is then used to calculate the MQE_m, which is the average value of every neuron's mqe_n for the given SOM layer. If $MQE_m \geqslant \tau_m \times mqe_0$, a new row or a new column of neurons will be inserted to this SOM, where τ_m is a parameter which controls the width of the SOM layer when training.

After adjusting the layer width, the algorithm creates a new layer with new SOMs, one for every neuron in the previous layer that has $mqe_i < \tau_\mu \times mqe_0$, where τ_μ is a parameter which controls the depth of the model.

The higher τ_m, the more tolerance, producing less clustered groups, while higher τ_μ translates to more tolerance which produces fewer layers and vice versa.

2.3.6 Neural Network with Multi Linear Regression

Reinforcement learning in the form of neural networks (NNs) has also been used for external energies disturbances on the speed profile of a driving profiles so that it could then be combined with multiple linear regression (MLR) for the estimation (De Cauwer *et al.*, 2017), using *EVteclab*'s dataset (Section 2.2). The neural network was responsible for the speed profile prediction, supplying two additional predictors (Constant motion factor (CMF) and aerodynamic factor (AF)) to the MLR. The MLR would then predict the energy consumption for the vehicle, and not the range, as depicted on the Figure 2.13.

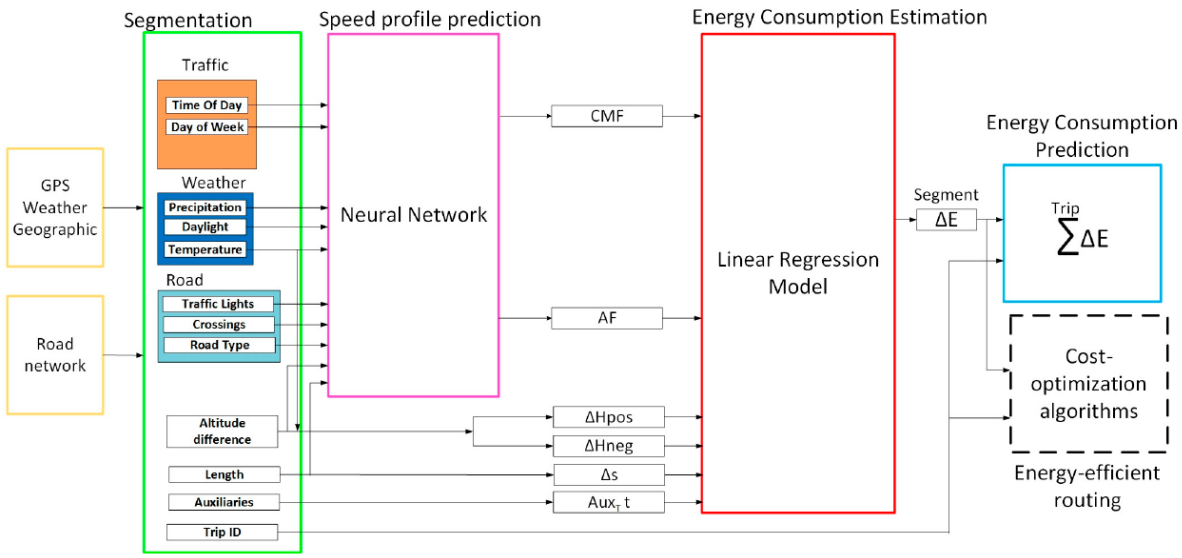

Figure 2.13: Neural Network + Multiple linear regression model (from De Cauwer *et al.*, 2017).

2.4 Summary

This chapter explored some core concepts required for the eRange prediction problem, namely energy storage; the SOC inference through the Coulomb counting method and the "Basic" eRange prediction from these parameters. Then introduced machine learning concepts such as type of learning (supervised, unsupervised and reinforcement), as well as train data overfitting, balance and variance.

The next section was dedicated to the dataset requirements for the eRange prediction. The existence of these datasets allows for estimation algorithms to be tested and in machine learning algorithms, trained.

Finally, the chapter ended with noteworthy eRange estimation algorithms. Starting with the non machine learning adaptive "History-based" algorithm and followed by machine learning techniques and their variants, such as ensemble learning, DTs, gradient boosting, SOMs and neural networks.

Although more complex than non machine learning solutions, the use of machine learning for the eRange estimation problem had reduced the prediction error, and thus further justifying its usage in this project for the eRange prediction problem. The following Table 2.2 summarizes the machine learning type, algorithms and datasets.

Table 2.2: Machine learning eRange prediction approaches as detailed on Sections 2.2, 2.3.

Implementation	ML Algorithms	Training datasets
Ullah *et al.*, 2021	ESG (DT, RF, KNN)	JARI
Zhao *et al.*, 2020	XGBoost + LightGDM	NDANEV
Lee and Wu, 2015	NN + SOM	N / A
Zheng *et al.*, 2016	SOM + RT	N / A
De Cauwer *et al.*, 2017	NN + MLR	EVteclab

3

Proposed Approach

This work aims to provide an eRange prediction machine learning model based on the best performing sub-model when training from publicly available datasets. To this end, a dataset was constructed on the preprocessing phase of this project from two publicly available datasets *VED* (Oh *et al.*, 2019); *ChargeCar* (*ChargeCar Database* n.d.) as detailed on Section 3.2. Section 3 describes how this dataset is then used for training by two machine learning algorithms: Ensemble stacked generalization (Ullah *et al.*, 2021) and Linear regression, the former with two distinct configurations. Each of the sub-models follow the standard supervised machine learning two stage process of first learning the dataset features with the expected eRange result (fitting). Once these machine learning algorithms are trained, all algorithms and then used for the prediction of eRange through obtained from real-time parameters. One of the models is then selected based on its performance, through different performance metrics (Section 3.3) and used for future execution in a real-time trip as detailed in Figure 3.1.

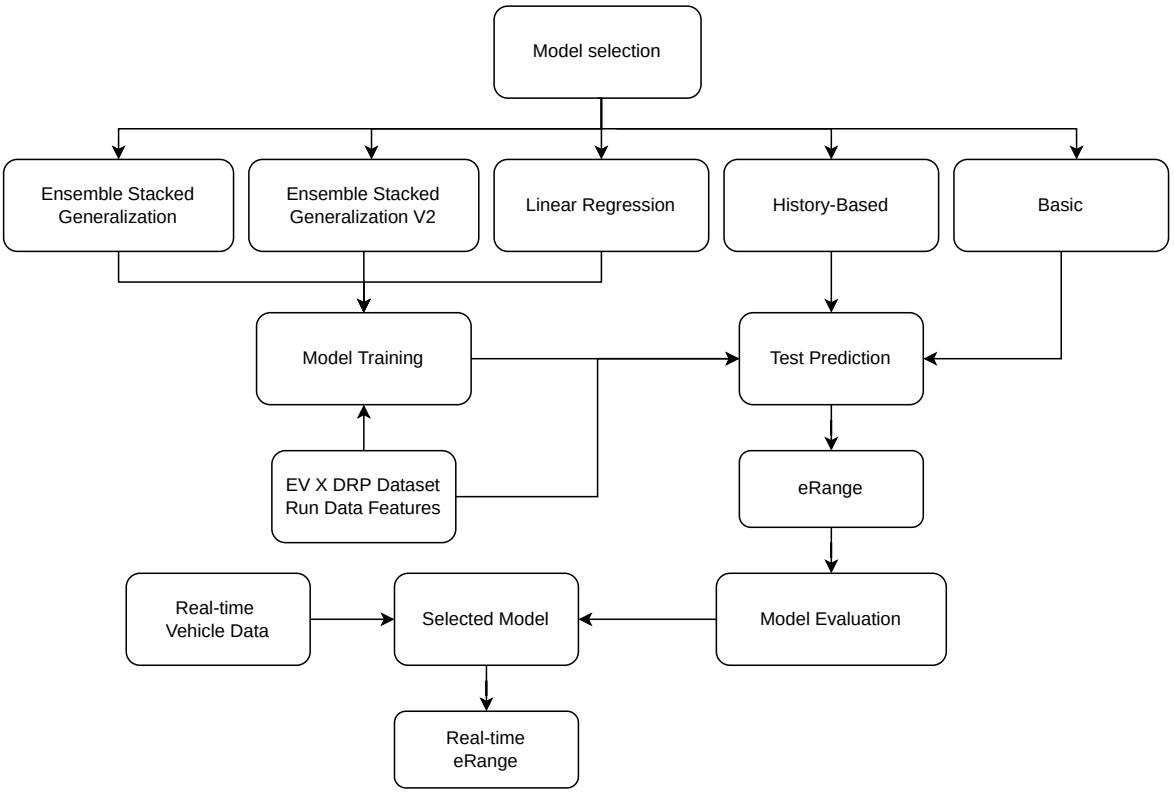

Figure 3.1: Project's system overview.

3.1 Methodology

This project addresses the eRange estimation problem through the use of a machine learning based model, being comprised by three distinct phases: the Dataset preprocessing phase, the Learning phase and the Estimation phase.

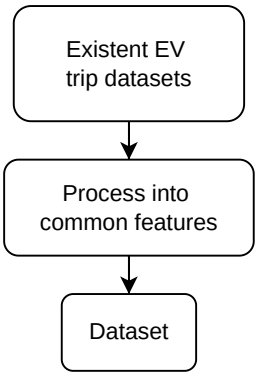

Figure 3.2: System overview - Dataset preprocessing.

On the Dataset preprocessing phase (Figure 3.2) a dataset will be created from historical traffic data recorded vehicle trips, as well as external existing and publicly available datasets from both the *VED* (Oh *et al.*, 2019) and *ChargeCar* (*ChargeCar Database* n.d.) integrated into the project's dataset. The resulting dataset contains multiple trips from different EVs, containing their respective static battery information provided by the manufacturer and multiple monitoring parameters in a time series format.

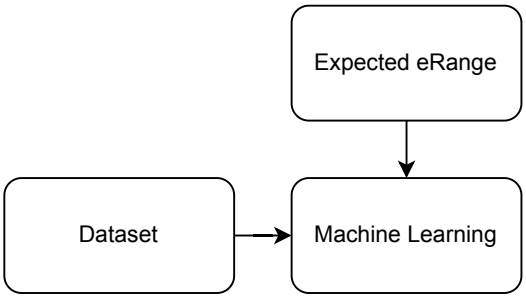

Figure 3.3: System overview - Learning phase.

The generated dataset will then be used to train the selected eRange prediction models on the Learning phase through machine learning, allowing it to fit its eRange estimation for each trip on the dataset as depicted in Figure 3.3. The expected eRange is supplied by an implementation of an eRange estimation *"History-based"* algorithm (Coutinho, 2021b) as detailed on Section 2.3.1.

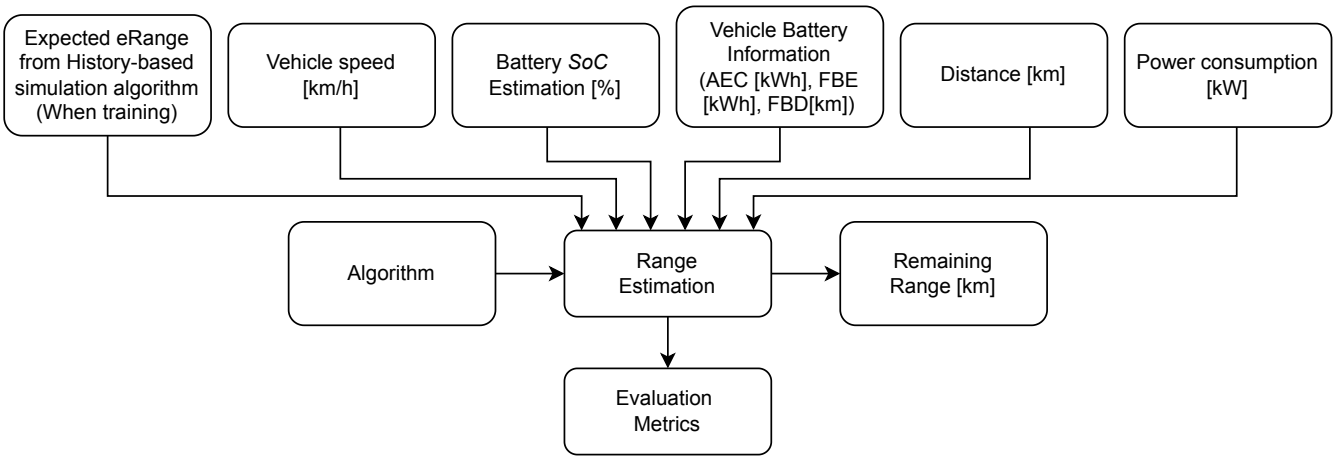

Figure 3.4: System overview - Estimation phase.

After training the machine learning algorithms with the dataset, the Estimation phase performs the eRange prediction on live SOC monitoring of a driving EV (Figure 3.4). The resulting prediction is then used for the calculation of the evaluation metrics so

than it can be compared with other eRange prediction algorithms (as described in Section 3.3).

The application features training configurations such as dataset feature configuration, minimum trip time and trip minimum time step, while also providing execution configurations, as prediction algorithms and evaluation methods.

As future integration with the Coutinho, 2021a which aims to transform a 1993 Rover Mini Cooper 1.3i (1300cc) into a fully EV, has been set as a goal for future integration of this project, a Python application was developed.

Through this integration, the vehicle could then request a new eRange estimation with real-time battery and road information through the existing eMini project's software. The application features configuration for the project's dataset preprocessing as well as machine learning model training: *training datasets*; *Minimum time-series time-step* controlling the minimum time interval for when datasets don't have it fixed; *Minimum trip execution time* to help reduce training bias to trips with lower execution times and execution trip.

The ensemble stacked generalization model was chosen as one of our machine learning algorithms due to its configuration being available for replication on its paper (Ullah *et al.*, 2021), thus the ensemble stacked generalization implemented on this work follow its configurations.

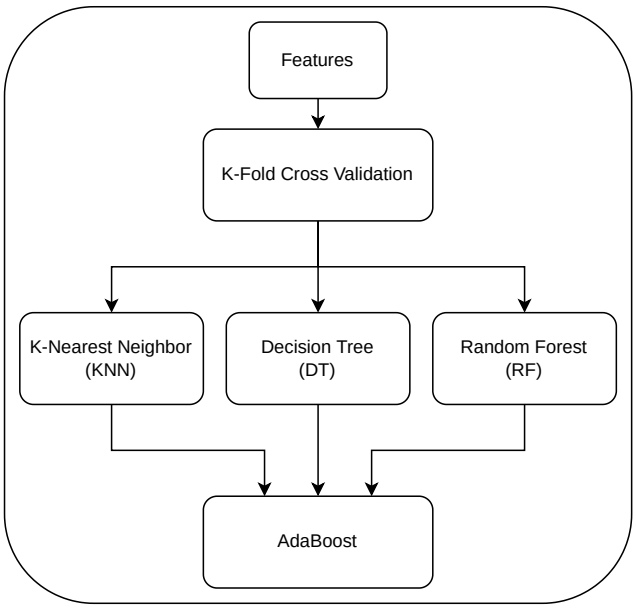

Figure 3.5: Ensemble Stacked Generalization model (adapted from Ullah *et al.*, 2021).

The paper's implementation however, differences in this model as the model's original

application was the EVs energy consumption prediction and not eRange, as well as the lack of availability of its *JARI* dataset (in 2.2) which made its implementation more challenging integration with other datasets.

In the DT algorithm, number of leafs is set to 4782, the number of nodes however, could not be set to 9563 due to the *scikit-learn* Python library's constraints. For both the DT and RF, depth is set at 15 and the maximum number of features NF (Liu *et al.*, 2019), is given by

$$NF = \sqrt{M}\,, \tag{3.1}$$

where M denotes the total number of features present in the training data. RF has a minimum number of trees set to 65, the K nearest neighbors algorithm uses the euclidean distance metric and K=10, as with Stratified KFold, K=10. Finally, the AdaBoost configuration has a learn rate of 1.0, number of estimators set to 100, the learning algorithm follows the exponential function algorithm (SAMME.R) and linear loss function.

An additional ESG implementation named *ESG V2* was derived from the original ensemble stacked generalization approach, maintaining the underlying machine learning algorithms, but changing some configurations to better fit the project's constructed dataset. The maximum number of features configured for decision tree and random forest algorithms are 9 and 7 respectively. As for the K-Nearest neighbor, K is configured at 70, the distance metric is *minkowski* with paremeter p set to 1.

Multiple Linear Regression was chosen for the second machine learning, due to its lower complexity and easier implementation, following the standard configuration defined by the *scikit-learn* python library.

As the eRange prediction problem of determining the vehicle's maximum distance is approached in a supervised learning implementation, the expected eRange is required to train the model. This however was an obstacle due to the real eRange not being present on the datasets chosen for this project. To solve this issue, the *"History-based" adaptive algorithm* in Coutinho, 2021b was chosen for the training phase's target for the estimators, as it did not use machine learning for its deemed optimistic results and less frequent updates. This approach estimates real-time AEC according to a 10 minute sliding observation window of the trip's instantaneous energy consumption, as well as real-time SOC value. This algorithm was designed as better alternative to its *"Basic" algorithm* in which the estimation uses the vehicle's (constants) AEC and FBD provided by the manufacturer and a real-time SOC value, and thus, was also included as comparable algorithm.

3.2 Data Collection

When training machine learning model for regression problems, the accuracy of the results on test data for different vehicles will depend on the diversity of the data used in training, namely, the higher vehicle information and trip diversity are present, the more likely it will reduce overfitting on our model with a single vehicle type as well as a single consumption profile. As the time-series presents static battery information which can be associated with a specific battery consumption, machine learning models can use this information to better predict the vehicle's unique consumption profile.

Most of these datasets are typically reserved for the EV manufacturers, as eRange is a competitive feature of these vehicles. Because of this, it was opted for publicly available datasets as they featured no fee and were eventually accessible.

To ensure model effectiveness on different vehicles, we opted for a diverse EV model dataset built from existing available datasets *VED* (Oh *et al.*, 2019); *ChargeCar* (*ChargeCar Database* n.d.) *Classic EV X Project DRP* (Coutinho, 2021b) as depicted on Figure 3.6.

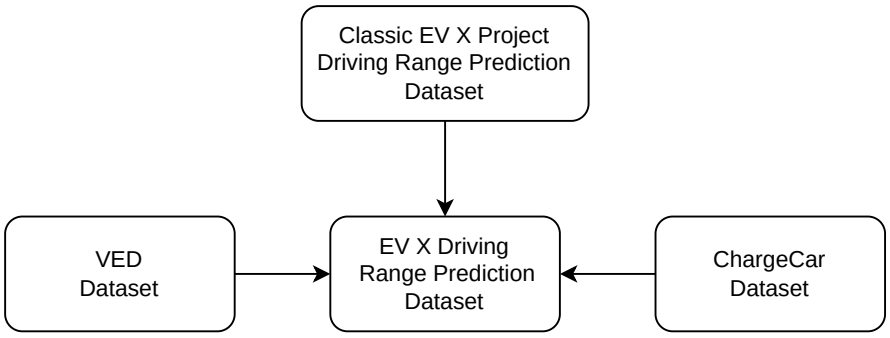

Figure 3.6: EV X Driving Range Prediction dataset sources.

The algorithm integrates EV trip datasets for training, thus requiring EV trips time-series with the following features: SOC (percentage), power consumption (kWh), distance (km) and speed (km/h), as well as vehicle information: AEC (kWh), FBE (kWh), and Full battery distance (FBD) (km) (as this information is static it is repeated in the time-series). For this reason, both *VED* (Oh *et al.*, 2019), *ChargeCar* (*ChargeCar Database* n.d.) and *Classic EVX* (Coutinho, 2021b) datasets were chosen. When configuring the algorithm's training, different datasets can be selected, as well as a minimum trip type and minimum driving time, as these variables have been tested and found to highly influence ML methods performance.

During the preprocessing phase, some features such FBE, FBD and AEC are sometimes not available on certain datasets. These however can be obtained from existing static EV information datasets as *Electric Vehicle Database* n.d. Other features such as power variation and distance, are trip dependent and must therefore be calculated.

The acceleration a is needed to calculate the distance feature of the dataset and for its calculation, the difference of the two trip instance velocity values is divided by the time variation Δt, then

$$a = \frac{v_f - v_i}{\Delta t} . \tag{3.2}$$

For the distance ΔD computation, we take into account the previously calculated acceleration a between trip instants and apply it to the initial velocity for the time variation, that is

$$\Delta D = v_i \times \Delta t + \frac{1}{2} \times a \times \Delta t^2 . \tag{3.3}$$

The constructed dataset is mainly composed by short vehicle trips of less than 20 minutes, presumably from short city commutes. This disparity in the training data could cause imprecise prediction on longer trips where different consumption profiles are observed, such as traveling on a highway. Newer datasets implemented in the future with longer EV trips, as well as the eMini project (Coutinho, 2021a) integration for newer trip monitoring could mitigate this issue on the future.

3.3 Evaluation Metrics

As prediction accuracy must be measured for each eRange prediction algorithm, the K-Fold cross validation method is used for determining the models' fitness. As described in the previous chapter, this cross validation method prevents validation bias by splitting the dataset into K folds, and testing each fold against the others for testing. With this method, five evaluation metrics are then chosen for the model's evaluation: *Mean absolute error (MAE) Mean squared error (MSE)*, *Mean absolute percentage error (MAPE)*, *Root mean squared error (RMSE)* and *Coefficient of determination (R2)*, as demonstrated in formulas 4 to 7 ($\mathbf{y_i}$ represents observed value, $\hat{\mathbf{y}}_\mathbf{i}$ the predicted value and $\bar{\mathbf{y}}_\mathbf{i}$ the average observed value) (Chicco *et al.*, 2021). As cross validation method returns the same metric K times, the average values for each metric are then used for prediction model comparison.

MAE represents the average of the absolute difference between the actual and predicted values, increasing weights for values closer to the average, mitigating outliers effects on this metric, is defined by

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \ . \tag{3.4}$$

MAPE indicates the same information as MAE, however, makes it more clear to compare between models due to its value is represented in a percentage. Unfortunately, this formula has complications for when the observed value is close to zero, due to the division by error, as shown:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}|\frac{y_i - \hat{y}_i}{y_i}| \times 100\% \ . \tag{3.5}$$

MSE measures both bias and variance of the residuals, averaging the squared difference between the original and predicted values, and is defined by

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \ . \tag{3.6}$$

RMSE measures the standard deviation of residuals with the square root of MSE, as defined by

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \ . \tag{3.7}$$

Lastly R2 represents the proportion of the variance in the dependent variable, and contrary to other the other evaluation metrics, where lower values are perceived as better, higher values for R2 are desirable, as lower values indicate redundant or inconsequential variables, calculated by

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \ . \tag{3.8}$$

In this work, the prediction model is compariatably considered more accurate when it has lower MAE, MAPE, MSE, RMSE values and a higher R2 value.

3.4 Developed Application

The focus of the application is to allow the comparison of existing eRange prediction models with the project's implemented model. For this reason, three additional prediction models (besides *"Basic"* and *"History-based"* from Coutinho, 2021b) were integrated into the application: ensemble stacked generalization approach of Ullah *et al.*, 2021, a modified ensemble stacked generalization approach with an adjusted configuration to better suit the training dataset and multiple linear regression.

From this work, a software application was developed, aiming for a future integration with the Coutinho, 2021a project which aims to transform a 1993 Rover Mini Cooper 1.3i (1300cc) into a fully EV.

Because the software would have to later be installed on a RaspberryPi device for vehicle communication, two programming languages were available for implementation: C and Python. The popularity amongst Python adoption as a programming language cemented this choice as it would benefit the learning curve for future developers to study with it.

The developed application features algorithm training and trip execution customization settings. The selection of enabled eRange prediction algorithms for training allows multi algorithm comparison for the same test trip.

Some limitations exist when using the *"Basic"* and the adaptive *"History-based"* eRange estimation algorithms from the *Classic EV X Project Driving Range Prediction* for machine learning training, as both algorithms require the EV model's AEC and FBE provided by the manufacturer, limiting the training to datasets that do provide this information, effectively excluding datasets such as the NDANEV.

4

Experimental Evaluation

As described in Section 3, the constructed dataset features multiple EV trips, most of them being shorter city comutes, differ in consumption of traveling in highways on longer trips. This can cause training bias with worse predictions on longer trips due to the reduced samples in the training data.

To study this effect, a longer vehicle trip was chosen for testing while the remaining trips were used for training, defining Minimum trip time (MTT), a minimum time required for a trip to be allowed into the training set of the model.

The MTTs chosen for this study were of 0, 10, 20, 30 and 40 minutes respectively and for each MTTs. This limitation reduces the number of trips used for training, the higher the value is, reducing the training time as it will be later demonstrated. The eRange prediction will be calculated for the selected test trip, while the time and evaluation metrics for each of them, as well as the KFold cross validation results being summarized at the end of this chapter on Tables (4.7 and 4.8) .

As stated on Chapter 3, the KFold cross validation method prevents validation bias by splitting the dataset into K folds. For the following results, a value of K=20 was chosen, testing 1/20 of the dataset on each fold.

For the trip execution, minimum trip limit values (10m, 20m, 30m, 40m) which would limit the selected trips for training was set. The reason behind this choice was due to the observed training time would be reduced, although results with the training limitation would fare worse on all evaluation metrics.

For single execution testing, where machine learning algorithms were trained with all trips except the test one, the implemented eRange prediction algorithms (*"Basic"*, *"History-based"*, Linear Regression (LR), Ensemble Stacked Generalization (ESG) and ESG V2) were selected for execution for the VED dataset's trip `E1/VED_171213_week_772_455-AC_ON.csv`.
This trip was generated in the preprocessing phase through the filtering of the EV dataset identifier 455 which belongs to 2013 Nissan leaf with trip identifier 171213. This configuration limits the dataset usage of these results to the *VED* dataset to avoid possible data discrepancies from other EVs present on external datasets.

Computation Platform Details: The computer operative system is Manjaro Linux with a 5.18.19-3-MANJARO kernel, the Python runtime is 3.9, hardware specifications for the CPU: AMD Ryzen 9 3900X (24) @ 3.800GHz and RAM: 48Gb RAM.

The application displays different eRange prediction results for the selected trip and prediction algorithms, allowing for an easy overview of the different dataset parameters, allowing initial input dataset configuration to depend on multiple datasets.

A conventional 47 minutes trip from the VED dataset (Oh *et al.*, 2019) for a Nissan Leaf model (2013) was selected for testing, with training data from the same dataset with a minimum trip time of 10 minutes and a timestamp of 0 seconds. The tested EV trip's state of charge (SOC), speed and instantaneous energy consumption (IEC) can be seen on Figures 4.1, 4.2 and 4.3 respectively.

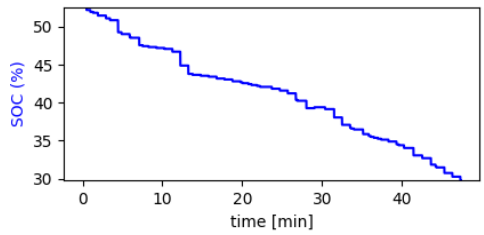

Figure 4.1: Test trip state of charge.

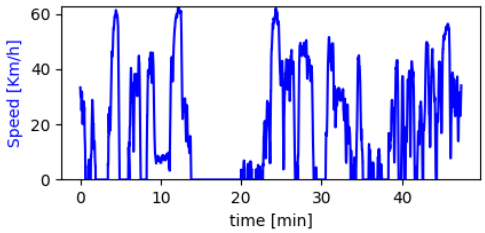

Figure 4.2: Test trip speed.

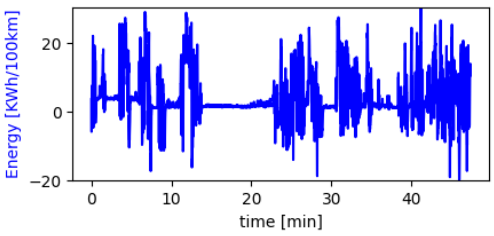

Figure 4.3: Test trip instantaneous energy consumption

4.1 Non machine learning eRange predictions

The comparison of the different selected non machine learning eRange prediction models can be seen on the following Figure 4.4, with predictions for the previously selected testing trip. The implemented non machine learning algorithms here demonstrated are: *"Basic"* and *"History-based"* approaches from (Coutinho, 2021b).

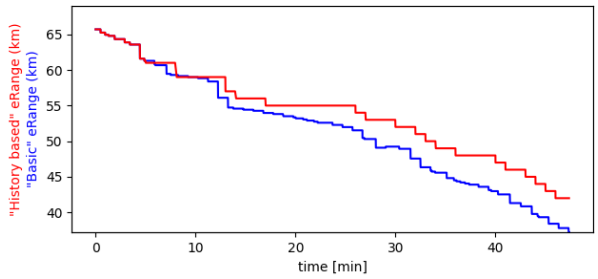

Figure 4.4: Test trip eRange prediction for *"Basic"* and *"History-based"* approaches.

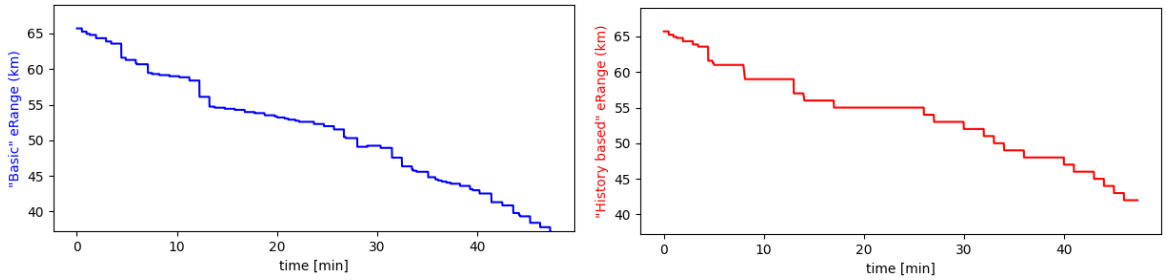

Figure 4.5: Test trip eRange prediction for *"Basic"* and *"History-based"* approaches (isolated).

In Figure 4.4, the *"History-based"* approach shows "plateau" sections when the minimum instance energy is not enough to trigger a recalculation for the eRange. It is worth noting the initial resemblance with the *"Basic"* approach from the same paper in which is *"History-based"* approach is based off.

4.2 Machine learning eRange predictions

4.2.1 Training impact of dataset configurations

Because machine learning algorithms are sensitive to the training data, it was decided to repeat the trip testing with five different training data configurations.

This execution was repeated five times, for each time, the minimum trip time (MTT) required for the training dataset would increase by 10 minutes. Doing so, the number of available trips would decrease, limiting training with longer trips and therefore, more similar consumption profiles to the selected testing trip. The effects of limiting the MTT on the number of trips selected for training can be seen on Figure 4.6.

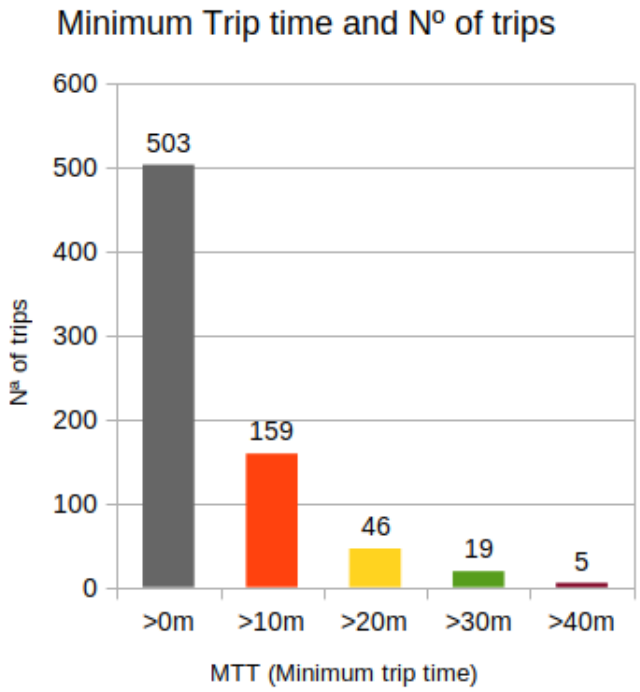

Figure 4.6: Visual minimum trip time (MTT) representation, demonstrates the amount of trips that are filtered by setting MTT.

The time results for the executions can be observed on the Table 4.1 bellow, as well as cross validation times in Table 4.2, while a minimum 10 minute limit execution and an unlimited execution will be presented on Sections 4.2.3 and 4.2.2.

Table 4.1: The effect of the minimum trip time (MTT) limitation on machine learning training time with the dataset and testing time with selected trip for ESG original paper configuration Ullah *et al.*, 2021; ESG V2 adjusted for dataset configuration and LR in purple.

ML approach	MTT	Trip count	Train time	Avg test time	All test time
LR	0m	503	138ms	0ms	4,223ms
ESG	0m	503	7m 25s 772ms	9ms	45s 020ms
ESG V2	0m	503	14m 10s 940ms	9ms	45s 300ms
LR	10m	159	72ms	0ms	4s 129ms
ESG	10m	159	4m 2s 146ms	9ms	44s 150ms
ESG V2	10m	159	7m 37s 998ms	9ms	44s 308ms
LR	20m	46	29ms	0ms	4s 243ms
ESG	20m	46	1m 33s 534ms	9ms	44s 841ms
ESG V2	20m	46	2m 52s 129ms	9ms	45s 127ms
LR	30m	19	14ms	0ms	4s 143ms
ESG	30m	19	41s 343ms	9ms	44s 158ms
ESG V2	30m	19	1m 11s 669ms	8ms	42s 490ms
LR	40m	5	6ms	0ms	4s 108ms
ESG	40m	5	9s 706ms	6ms	32s 980ms
ESG V2	40m	5	15s 505ms	7ms	33s 510ms

In Table 4.1, are presented the three machine learning algorithms: Linear Regression (LR), the original Ensemble Stacked Generalization implementation from Ullah *et al.*, 2021 (ESG) and the Ensemble Stacked Generalization configured for our dataset (ESG V2). For each *minimum trip time (MTT)*, the *train time*, *avg test time* (average test time) and *all test time* are presented. The *train time* indicates the amount of time it takes to train the model; the *avg test time* indicates the average time it takes for the model to predict an eRange on a given instant and *all test time* is the amount of time it took to predict every instant of the selected test trip.

As expected, when increasing minimum trip time and therefore reducing the available trips, the training and testing times are significantly reduced.

Table 4.2: The effect of the minimum trip time (MTT) limitation on machine learning training K=20 Fold cross validation for ESG original paper configuration Ullah *et al.*, 2021; ESG V2 adjusted for dataset configuration and LR in purple.

ML approach	MTT	Trip count	All folds train time	Avg fold train time
LR	0m	503	30s 064ms	1s 503ms
ESG	0m	503	4h 3m 54s 919ms	12m 11s 745ms
ESG V2	0m	503	6h 50m 52s 398ms	20m 32s 619ms
LR	10m	159	14s 543ms	727ms
ESG	10m	159	1h 57m 29s 701ms	5m 52s 485ms
ESG V2	10m	159	3h 22m 59s 269ms	10m 08s 963ms
LR	20m	46	3s 805ms	190ms
ESG	20m	46	41m 38s 272ms	2m 04s 913ms
ESG V2	20m	46	1h 14m 30s 373ms	3h 43s 518ms
LR	30m	19	1s 143ms	57ms
ESG	30m	19	18m 14s 883ms	54s 744ms
ESG V2	30m	19	30m 39s 978ms	1m 31s 998ms
LR	40m	5	230ms	11ms
ESG	40m	5	5m 26s 732ms	16s 336ms
ESG V2	40m	5	8m 40s 895ms	26s 044ms

In Table 4.2, are presented the same three machine learning algorithms, this time reporting the K=20 fold cross validation time metrics.

The *all folds train time* indicates the total time for all folds to train the model. Note that we were able to use CPU parallelization on each fold, the average time it takes for each fold to complete is much smaller, as represented by the *Avg fold train time* (average fold train time).

Unfortunately, we were not able to retrieve testing time for all folds as well as the average fold testing time. This was due to the fact that the used Python library (scikit-learn) either did not make it available nor is documented. As cross validation was run in a parallel manner to speed up results utilizing the maximum number of CPU cores, it was not possible to infer the total testing time for all folds, only the time it took for all folds to complete asynchronously which was decided to not be included on this study due to the lack of relevancy.

4.2.2 Unlimited Training Execution

When training the machine learning algorithms on this section, the minimum trip time required for a trip to be eligible for training is not set. This will not attempt to contradict the discrepancy in shorter EV trips present on our dataset.

Prediction of the eRange with the implemented machine learning algorithms as well as with the baseline "History-based approach" can all be seen on Figure 4.7, the baseline (in red), ensemble stacked generalization (in green), ensemble stacked generalization configured for our dataset (in light blue) and linear regression (in purple).

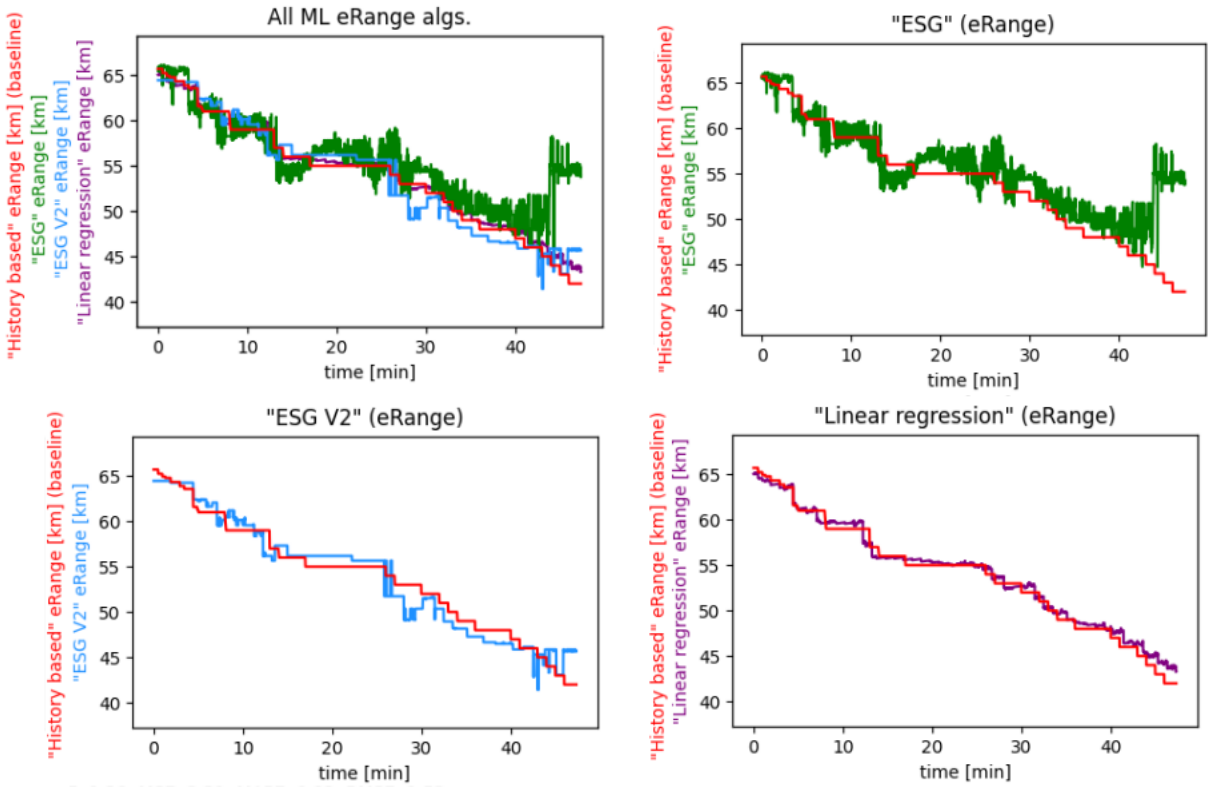

Figure 4.7: All ML eRange predictions for ESG original paper configuration in green Ullah *et al.*, 2021; ESG V2 adjusted for dataset configuration in light blue; and LR in purple (for training with no minimum trip duration).

As we can observe on Figure 4.7, the linear regression fared better in following the near linear trend of the baseline "History-based approach" algorithm. The ensemble stacked generalization fared worse as it followed the configurations specified on Ullah *et al.*, 2021, while our small adjustments of ensemble stacked generalization (V2) resulted in less prediction variance.

The machine learning algorithms' prediction metrics for this trip are presented on Table 4.3, the KFold cross validation metrics are presented on Table 4.4. Note that in metrics such as MAE, MSE, MAPE, RMSE, the lower the value, the better however, the $R2$ value is preferred the closest to 1.

Table 4.3: Ensemble Stacked Generalization and Linear Regression prediction metrics (for training with no minimum trip duration).

ML approach	**MAE**	**MSE**	**MAPE**	**RMSE**	**r^2**
LR	**0.630**	**0.631**	**0.012**	**0.795**	**0.984**
ESG	2.614	17.173	0.055	4.144	0.560
ESG V2	1.202	2.314	0.023	1.521	0.941

The observed results seen on Table 4.3 show a better predictive performance for linear regression, followed by the adaptation on the ensemble stacked generalization (ESG V2). The original ensemble stacked generalization (ESG) suffers the most on the $R2$ prediction metric, indicating the model fits the data poorly in relation to the testing data, in this case, the type of smaller EV trips trained are significantly different from the longer tested trip.

The LR algorithm improves on the *"History-based"* approach that exhibits a "stair-case effect", which may cause anxiety on the driver, each time the estimated value drops in a step. Thus, LR seems to be the best ML approach addressed so far. On the other hand, the ESG algorithm provides an initial pessimistic, followed by more optimistic estimates, yielding larger eRange predicted values. One possible cause for this performance may be the missing features from the original dataset training such as elevation, however the lower value of R2 also indicates poorer fitting between the selected dataset features and the prediction value.

Table 4.4: Ensemble Stacked Generalization and Linear Regression cross validation prediction metrics (for training with no minimum trip duration).

ML approach	MAE	MSE	MAPE	RMSE	r^2
LR	**0.539**	**0.725**	**0.341**	**0.807**	**0.998**
ESG	1.479	4.724	1.156	2.106	0.991
ESG V2	1.279	2.996	1.106	1.677	0.994

The K=20 cross validation metrics presented in Table 4.4 show that even though ESG scored an r^2 of 0.56, was not good for this trip, it fared a lot better for the rest of the dataset when other trips were selected for testing. In general, the average r^2 of ESG was 0.991, meaning that for our selected testing trip was an outlier in our dataset.

This was expected, since the most EV trips in our dataset are small, and therefore incentivize further training with longer trips, thus avoiding overfitting the model to shorter trips.

4.2.3 Limited Training Execution

Here it is discussed the effects of limiting the minimum training time required for each trip to be eligible to train our machine learning algorithms. The goal of this section is to describe the effects of imposing the minimum trip time (MTT) of 10 minutes on the EV trips that are used for training, avoiding overfitting the models to shorter trips.

The trip that was previously tested on the prior section is now tested with MTT=10 minutes. Figure 4.8 shows the eRange baseline from the non machine learning "history-based approach" (in red); ensemble stacked generalization (Ullah *et al.*, 2021) (in green), ensemble stacked generalization V2 (in light blue); and the linear regression approach (in purple). The evaluation metrics are reported in Tables 4.5 and 4.6.

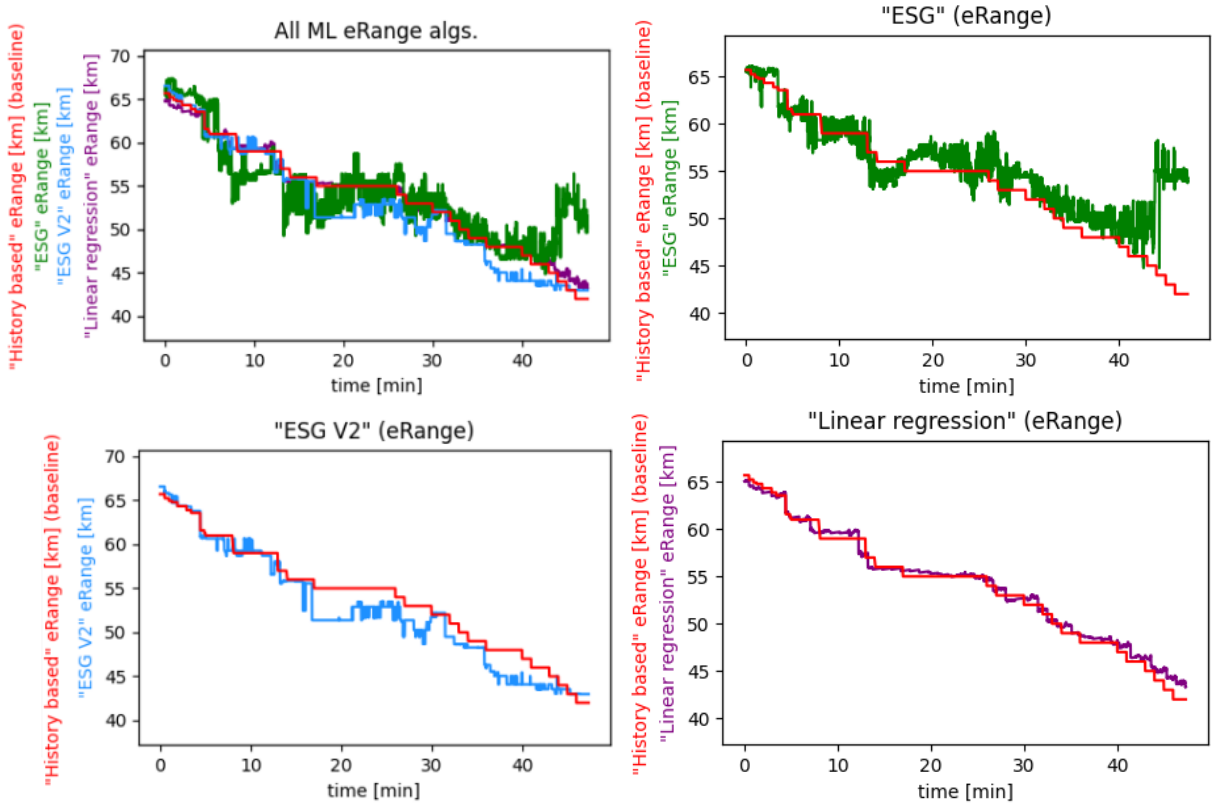

Figure 4.8: Isolated ML eRange predictions for ESG original paper configuration in green Ullah *et al.*, 2021; ESG V2 adjusted for dataset configuration in light blue; and LR in purple (for training with trips above or equal to 10 minute duration).

Table 4.5: Ensemble Stacked Generalization and Linear Regression prediction metrics (for above or equal to 10min trip training).

ML approach	MAE	MSE	MAPE	RMSE	r^2
LR	**0.597**	**0.603**	**0.012**	**0.776**	**0.985**
ESG	2.317	12.135	0.047	3.483	0.689
ESG V2	1.519	3.650	0.029	1.910	0.906

As we can observe, the imposed minimum trip time benefitted both LR ($r^2 = 0.984$ to 0.985, $+0.1\%$) and ESG ($r^2 = 0.560$ to 0.689, $+23\%$) while harming ESG V2 accuracy ($r^2 = 0.941$ to 0.906, -3.71%). While ESG saw the most positive change on all metrics, even so, it is still not an ideal value, comparing with the other models.

Table 4.6: Ensemble Stacked Generalization and Linear Regression cross validation prediction metrics (for above or equal to 10min trip training).

ML approach	MAE	MSE	MAPE	RMSE	r^2
LR	**0.696**	**1.077**	**0.483**	**0.984**	**0.997**
ESG	1.687	5.833	1.319	2.288	0.987
ESG V2	1.339	3.343	1.051	1.778	0.991

Although MTT=10 had helped ESG on this test trip, the cross validation metrics indicate a slightly worse performance on all models when testing the rest of the database: LR ($r^2 = 0.998$ to 0.997, -0.1%) and ESG ($r^2 = 0.991$ to 0.987, -0.4%) while harming ESG V2 accuracy ($r^2 = 0.994$ to 0.991, -0.3%). These results indicate that limiting MTT=10m only benefitted ESG on this (longer) trip.

The following Tables 4.7 and 4.8 summarize the prediction metrics for both the selected trip and the KFold cross validation values respectively. The minimum trip time is represented by the MTT header.

Table 4.7: Ensemble Stacked Generalization and Linear Regression prediction metrics for all minimum trip times (represented by the MTT).

ML approach	MTT	MAE	MSE	MAPE	RMSE	r^2
LR	0m	0.630	0.631	0.012	0.795	0.984
ESG	0m	2.614	17.173	0.055	4.144	0.560
ESG V2	0m	1.202	2.314	0.023	1.521	0.941
LR	10m	0.597	0.603	0.012	0.776	0.985
ESG	10m	2.317	12.135	0.047	3.483	0.689
ESG V2	10m	1.519	3.650	0.029	1.910	0.906
LR	20m	0.581	0.589	0.011	0.767	0.985
ESG	20m	3.883	30.721	0.079	5.543	0.212
ESG V2	20m	2.184	7.525	0.043	2.743	0.807
LR	30m	0.706	0.785	0.013	0.886	0.980
ESG	30m	9.281	142.918	0.174	11.955	-2.664
ESG V2	30m	4.664	37.709	0.084	6.141	0.033
LR	40m	0.905	1.479	0.017	1.216	0.962
ESG	40m	7.179	95.455	0.139	9.770	-1.447
ESG V2	40m	5.115	31.937	0.101	5.651	0.181

Table 4.8: Ensemble Stacked Generalization and Linear Regression cross validation prediction metrics for all minimum trip times (represented by the MTT).

ML approach	MTT	MAE	MSE	MAPE	RMSE	r^2
LR	0m	0.539	0.725	0.341	0.807	0.998
ESG	0m	1.479	4.724	1.156	2.106	0.991
ESG V2	0m	1.279	2.996	1.106	1.677	0.994
LR	10m	0.696	1.077	0.483	0.984	0.997
ESG	10m	1.687	5.833	1.319	2.288	0.987
ESG V2	10m	1.339	3.343	1.051	1.778	0.991
LR	20m	0.944	1.805	0.672	1.262	0.980
ESG	20m	2.394	12.414	1.741	3.233	0.936
ESG V2	20m	1.712	5.425	1.367	2.213	0.964
LR	30m	1.167	2.807	1.040	1.461	0.934
ESG	30m	3.805	30.432	3.033	4.692	0.193
ESG V2	30m	2.835	14.028	2.479	3.390	0.510
LR	40m	1.559	5.156	1.488	1.914	-1.619
ESG	40m	3.265	25.787	2.910	4.178	-6.441
ESG V2	40m	2.877	14.416	3.010	3.430	-6.885

These experimental results show that eRange prediction can be achieved with machine learning techniques, overcoming the existing *"Basic"* and *"History-based"* approaches.

The LR model has fast training and achieves adequate results with a smooth varying curve on the prediction values.

After some configurations, the ESG V2 provided batter predictive results than ESG for this dataset. Machine learning prediction algorithms suffer on every metric when the minimum trip limit is increased, except on specific cases such as longer trips, which benefit the original ESG when 0>MTT>10. This suggests the need for increased dataset training size, event if longer trips are not present. Another possible reason for this worse predictive performance can be the lower number of longer trips for training.

The ESG models presents higher predictive errors with higher variance than the LR or the ESG V2 models, as well as for the selected testing trip, demonstrating the underlying fragility of the DTs in their tendency to be easily overfit to the training data. Further ways to reduce variance on the testing such as reducing number of features could increase the training bias (Liu *et al.*, 2019), and therefore mitigating overfitting.

5

Conclusions

In this thesis, we established the electric vehicles (EVs) and their rising popularity amongst consumers. As consumers suffer from range anxiety when not accurately knowing how much energy their vehicle has while driving, the eMini Project (Coutinho, 2021a) intends to improve the existing electric range (eRange) prediction (Coutinho, 2021b) from the known as the adaptive "History-based" approach.

As different approaches with machine learning promised better results for the eRange estimation problem, this project was dedicated to finding a suitable machine learning algorithm for later integration with the eMini Project.

The challenge of finding public domain datasets proved to be difficult, as most datasets would either be no longer accessible, state or company owned. Some datasets such as the VED (Oh *et al.*, 2019), ChargeCar (*ChargeCar Database* n.d.) and the ev-database (*Electric Vehicle Database* n.d.), were chosen for the project's machine learning algorithms training.

However, the expected eRange was not present in these datasets, difficulting the training process. Due to the fact that regression algorithms are supervised machine learning techniques, and require expected eRange for training, the adaptive "History-based" was elected as the baseline for the eRange estimation.

This work studied existing machine learning eRange prediction approaches where different algorithms were used, such as linear regression, ensemble stacked generalization, gradient boosted decision tree (GBDT), self-organizing maps (SOMs) and artificial neural networks. From these, linear regression and ensemble stacked generalization were implemented.

A Python application was developed, allowing for dataset preprocessing where combining datasets and engineering features which is required for training; algorithm configuration and testing; trip selection; visualization of dataset features as well as individual algorithm predictions.

An adjusted ensemble stacked generalization V2 model was created to contrast the changes required to maintain predictive accuracy when implementing models that were designed with different datasets in mind.

Finally, five tests on the same EV trip were demonstrated, showing different graphs for the first two executions and evaluating all five executions with standard metrics and cross validation to avoid training bias.

The results of using data from publicly available datasets showed that linear regression performed better out of the three models, not being sensitive to training as the stacked generalization implementation, which presented signs of overfitting to more commonly trained shorter trips.

The stacked generalization V2 performed better than the original (Ullah *et al.*, 2021), however it heavily faltered its accuracy when reducing the training set.

The resulting project has completed its goals of developing an application capable of using machine learning to improve on existing eRange prediction approaches.

From this work, two papers have been published for both the RECPAD2022 (Portuguese Conference on Pattern Recognition Albuquerque *et al.*, 2022a) and ICPRAM2023 (INSTICC International Conference on Pattern Recognition Applications and Methods Albuquerque *et al.*, 2022b).

5.1 Future Work

For further comparison options, additional machine learning approaches could be added, as well as more data from previously unaccessible datasets into the implemented Python application. We also plan to include more dataset preprocessing features, such as driving patterns, road elevation, vehicle load and location. The source code of the project is publicly available on the GitHub page (*Electric Vehicle X Driving Range Prediction Github repository* n.d.), promoting further contributions from the open source community.

As future work, we plan to perform the integration of this work with Classic eMini Project. The project features a combustion engine vehicle converted to battery electric. Our project is aimed at replacing the existing "history-based" eRange prediction algorithm, predicting with the real-time information of the EV to continuously provide better estimations as the model learns with the vehicle.

References

[1] Ji Zhu, Saharon Rosset, Hui Zou, and Trevor Hastie, "Multi-class adaboost", *Statistics and its interface*, vol. 2, Feb. 2006. DOI: 10.4310/SII.2009.v2.n3.a8.

[2] Murat Yilmaz and Philip T. Krein, "Review of battery charger topologies, charging power levels, and infrastructure for plug-in electric and hybrid vehicles", *IEEE Transactions on Power Electronics*, vol. 28, no. 5, pages 2151–2169, 2013. DOI: 10.1109/TPEL.2012.2212917.

[3] J. Xie, J. Ma, and K. Bai, "Enhanced coulomb counting method for state-of-charge estimation of lithium-ion batteries based on peukert's law and coulombic efficiency", *Journal of Power Electronics*, vol. 18, pages 910–922, May 2018. DOI: 10.6113/JPE.2018.18.3.910.

[4] Indranil Bose and Radha K. Mahapatra, "Business data mining - a machine learning perspective", *Information & Management*, vol. 39, no. 3, pages 211–225, 2001, ISSN: 0378-7206. DOI: https://doi.org/10.1016/S0378-7206(01)00091-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S037872060100091X.

[5] *Chargecar database*, http://www.chargecar.org/data, Accessed: 2022-07-03.

[6] *Chargecar's create rav4-recorder box*, http://chargecar.org/participate/lending, Accessed: 2022-07-03.

[7] David Coutinho, "Classic emini project: Electrification of a classic mini, technical report", Draft version, Jul. 2021. [Online]. Available: https://www.researchgate.net/publication/361823632_Classic_eMini_Project_

Electrification_of_a_Classic_Mini_TECHNICAL_REPORT_draft_version.

[8] David Coutinho, "Classic ev x project driving range prediction, technical report", Draft version, Jul. 2021. [Online]. Available: https://www.researchgate.net/publication/353210805_Classic_EV_X_Project_Driving_Range_Prediction_TECHNICAL_REPORT_draft_version.

[9] Achim Enthaler and Frank Gauterin, "Method for reducing uncertainties of predictive range estimation algorithms in electric vehicles", in *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, 2015, pages 1–5. DOI: 10.1109/VTCFall.2015.7391023.

[10] G. Schmitt, K. Moeller, and P. Plagemann, "Online monitoring of crevice corrosion with electrochemical noise", *Materials and Corrosion*, vol. 55, no. 10, pages 742–747, 2004. DOI: https://doi.org/10.1002/maco.200403812. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/maco.200403812. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/maco.200403812.

[11] Huixiang Liu, Qing Li, Bin Yan, Lei Zhang, and Yu Gu, "Bionic electronic nose based on mos sensors array and machine learning algorithms used for wine properties detection", *Sensors*, vol. 19, no. 1, 2019, ISSN: 1424-8220. DOI: 10.3390/s19010045. [Online]. Available: https://www.mdpi.com/1424-8220/19/1/45.

[12] Yang Song and Xianbiao Hu, "Learning electric vehicle driver range anxiety with an initial state of charge-oriented gradient boosting approach", *Journal of Intelligent Transportation Systems*, vol. 0, no. 0, pages 1–19, 2021. DOI: 10.1080/15472450.2021.2010053. eprint: https://doi.org/10.1080/15472450.2021.2010053. [Online]. Available: https://doi.org/10.1080/15472450.2021.2010053.

[13] Ona Egbue and Suzanna Long, "Barriers to widespread adoption of electric vehicles: An analysis of consumer attitudes and perceptions", *Energy Policy*, vol. 48, pages 717–729, 2012, Special Section: Frontiers of Sustainability, ISSN: 0301-4215. DOI: https://doi.org/10.1016/j.enpol.2012.06.009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0301421512005162.

[14] Carlos Gaete-Morales, Hendrik Kramer, Wolf-Peter Schill, and Alexander Zerrahn, "An open tool for creating battery-electric vehicle time series from empirical data, emobpy", *Scientific Data*, vol. 8, no. 1, page 152, 2021, ISSN: 2052-4463. DOI: 10.1038/s41597-021-00932-9. [Online]. Available: https://doi.org/10.1038/s41597-021-00932-9.

[15] Y. Zhang, W. Wang, Y. Kobayashi, and K. Shirai, "Remaining driving range estimation of electric vehicle", in *2012 IEEE International Electric Vehicle Conference*, ser. 2012 IEEE International Electric Vehicle Conference, 2012, pages 1–7. DOI: 10.1109/IEVC.2012.6183172. [Online]. Available: https://doi.org/10.1109/IEVC.2012.6183172.

[16] Martin Smuts, Brenda Scholtz, and Janet Wesson, "A critical review of factors influencing the remaining driving range of electric vehicles", in *2017 1st International Conference on Next Generation Computing Applications (NextComp)*, 2017, pages 196–201. DOI: 10.1109/NEXTCOMP.2017.8016198.

[17] Irfan Ullah, Kai Liu, Toshiyuki Yamamoto, Muhammad Zahid, and Arshad Jamal, "Electric vehicle energy consumption prediction using stacked generalization: An ensemble learning approach", *International Journal of Green Energy*, vol. 18, no. 9, pages 896–909, 2021. DOI: 10.1080/15435075.2021.1881902. eprint: https://doi.org/10.1080/15435075.2021.1881902. [Online]. Available: https://doi.org/10.1080/15435075.2021.1881902.

[18] Chung-Hong Lee and Chih-Hung Wu, "A novel big data modeling method for improving driving range estimation of evs", *IEEE Access*, vol. 3, pages 1980–1993, 2015. DOI: 10.1109/ACCESS.2015.2492923.

[19] Cedric De Cauwer, Wouter Verbeke, Thierry Coosemans, Saphir Faid, and Joeri Van Mierlo, "A data-driven method for energy consumption prediction and energy-efficient routing of electric vehicles in real-world conditions", *Energies*, vol. 10, no. 5, 2017, ISSN: 1996-1073. DOI: 10.3390/en10050608. [Online]. Available: https://www.mdpi.com/1996-1073/10/5/608.

[20] Davide Chicco, Matthijs J. Warrens, and Giuseppe Jurman, "The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation", *PeerJ Computer Science*, vol. 7, e623, Jul. 2021, ISSN: 2376-5992. DOI: 10.7717/peerj-cs.623. [Online]. Available: https://doi.org/10.7717/peerj-cs.623.

[21] Erik Figenbaum, Nils Fearnley, Paul Pfaffenbichler, Randi Hjorthol, Marika Kolbenstvedt, Reinhard Jellinek, Bettina Emmerling, G. Maarten Bonnema, Farideh

Ramjerdi, Liva Vågane, and Lykke Møller Iversen, "Increasing the competitiveness of e-vehicles in europe", *European Transport Research Review*, vol. 7, no. 3, page 28, 2015, ISSN: 1866-8887. DOI: 10.1007/s12544-015-0177-1. [Online]. Available: https://doi.org/10.1007/s12544-015-0177-1.

[22] Alessandro Brighente, Mauro Conti, Denis Donadel, and Federico Turrin, "Evscout2.0: Electric vehicle profiling through charging profile", *CoRR*, vol. abs/2106.16016, 2021. arXiv: 2106.16016. [Online]. Available: https://arxiv.org/abs/2106.16016.

[23] D. Albuquerque, A. Ferreira, and DPC Antão, "Estimating electric vehicle driving range with machine learning", in *INSTICC International Conf. on Pattern Recognition Applications and Methods - ICPRAM*, 2022, pages –.

[24] Xiao-Hui Sun, Toshiyuki Yamamoto, and Takayuki Morikawa, "Stochastic frontier analysis of excess access to mid-trip battery electric vehicle fast charging", *Transportation Research Part D: Transport and Environment*, vol. 34, pages 83–94, 2015, ISSN: 1361-9209. DOI: https://doi.org/10.1016/j.trd.2014.10.006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361920914001473.

[25] Xiao-Hui Sun, Toshiyuki Yamamoto, and Takayuki Morikawa, "Fast-charging station choice behavior among battery electric vehicle users", *Transportation Research Part D: Transport and Environment*, vol. 46, pages 26–39, 2016, ISSN: 1361-9209. DOI: https://doi.org/10.1016/j.trd.2016.03.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361920916000377.

[26] Kai Liu, Toshiyuki Yamamoto, and Takayuki Morikawa, "Impact of road gradient on energy consumption of electric vehicles", *Transportation Research Part D: Transport and Environment*, vol. 54, pages 74–81, 2017, ISSN: 1361-9209. DOI: https://doi.org/10.1016/j.trd.2017.05.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361920917303887.

[27] Kai Liu, Jiangbo Wang, Toshiyuki Yamamoto, and Takayuki Morikawa, "Exploring the interactive effects of ambient temperature and vehicle auxiliary loads on electric vehicle energy consumption", *Applied Energy*, vol. 227, pages 324–331, 2018, Transformative Innovations for a Sustainable Future Part III, ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2017.08.074. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261917310929.

[28] Tyson Condie, Paul Mineiro, Neoklis Polyzotis, and Markus Weimer, "Machine learning on big data", in *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, 2013, pages 1242–1244. DOI: 10.1109/ICDE.2013.6544913.

[29] Lina Zhou, Shimei Pan, Jianwu Wang, and Athanasios V. Vasilakos, "Machine learning on big data: Opportunities and challenges", *Neurocomputing*, vol. 237, pages 350–361, 2017, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2017.01.026. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231217300577.

[30] Saleema Amershi, Andrew Begel, Christian Bird, Robert DeLine, Harald Gall, Ece Kamar, Nachiappan Nagappan, Besmira Nushi, and Thomas Zimmermann, "Software engineering for machine learning: A case study", in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 2019, pages 291–300. DOI: 10.1109/ICSE-SEIP.2019.00042.

[31] Liang Zhao, Wei Yao, Yu Wang, and Jie Hu, "Machine learning-based method for remaining range prediction of electric vehicles", *IEEE Access*, vol. 8, pages 212 423–212 441, 2020. DOI: 10.1109/ACCESS.2020.3039815.

[32] *Machine learning-based method for remaining range prediction of electric vehicles - source*, https://github.com/liangzhao123/range_prediction, Accessed: 2022-07-18.

[33] B. Zheng, P. He, L. Zhao, and H. Li, "A hybrid machine learning model for range estimation of electric vehicles", in *2016 IEEE Global Communications Conference (GLOBECOM)*, ser. 2016 IEEE Global Communications Conference (GLOBECOM), 2016, pages 1–6. DOI: 10.1109/GLOCOM.2016.7841506. [Online]. Available: https://doi.org/10.1109/GLOCOM.2016.7841506.

[34] Andreas Rauber, Dieter Merkl, and Michael Dittenbach, "The growing hierarchical self-organizing map: Exploratory analysis of high-dimensional data", *Neural Networks, IEEE Transactions on*, vol. 13, pages 1331–, Nov. 2002. DOI: 10.1109/TNN.2002.804221.

[35] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu, "Lightgbm: A highly efficient gradient boosting decision tree", in *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

[36] T. Kohonen, "The self-organizing map", *Proceedings of the IEEE*, vol. 78, no. 9, pages 1464–1480, 1990. DOI: 10.1109/5.58325.

[37] Tom Mitchell, "The discipline of machine learning", Tech. Rep. CMU ML-06 108, 2006.

[38] *National big data alliance of new energy vehicles*, http://www.ndanev.com, Accessed: 2022-07-18.

[39] *Electric vehicle database*, https://ev-database.org/car/1011/Nissan-Leaf, Accessed: 2022-07-03.

[40] *Evteclab's eva platform*, http://proeftuin-ev.be/content/eva-platform, Accessed: 2022-07-19.

[41] *Jari research database*, https://www.jari.or.jp/research-database/, Accessed: 2022-07-03.

[42] *Project consigning technology development for rational use of energy (innovative manufacturing process technology development)*, https://web.archive.org/web/20130402003134/http://www.meti.go.jp/english/press/2012/0502_01.html, Accessed: 2022-07-17.

[43] *Paris agreement*, UN Treaty, United Nations, Dec. 2015. [Online]. Available: https://treaties.un.org/pages/ViewDetails.aspx?src=TREATY&mtdsg_no=XXVII-7-d&chapter=27&clang=_en.

[44] Bogdan Ovidiu Varga, Arsen Sagoian, and Florin Mariasiu, "Prediction of electric vehicle range: A comprehensive review of current issues and challenges", *Energies*, vol. 12, no. 5, 2019, ISSN: 1996-1073. DOI: 10.3390/en12050946. [Online]. Available: https://www.mdpi.com/1996-1073/12/5/946.

[45] Syed Haleem Shah, Yoseline Angel, Rasmus Houborg, Shawkat Ali, and Matthew F. McCabe, "A random forest machine learning approach for the retrieval of leaf chlorophyll content in wheat", *Remote Sensing*, vol. 11, no. 8, 2019, ISSN: 2072-4292. DOI: 10.3390/rs11080920. [Online]. Available: https://www.mdpi.com/2072-4292/11/8/920.

[46] D. Albuquerque, A. Ferreira, and DPC Antão, "Electric vehicle driving range prediction: An approach with machine learning", in *RECPAD Portuguese Conf. on Pattern Recognition - RecPad RECPAD*, 2022, pages –.

[47] M.K Yoong, Y.H Gan, G.D Gan, C.K Leong, Z.Y Phuan, B.K Cheah, and K.W Chew, "Studies of regenerative braking in electric vehicle", in *2010 IEEE Conference on Sustainable Utilization and Development in Engineering and Technology*, 2010, pages 40–45. DOI: 10.1109/STUDENT.2010.5686984.

[48] Ashwini Venkatasubramaniam, Julian Wolfson, Nathan Mitchell, Timothy Barnes, Meghan JaKa, and Simone French, "Decision trees in epidemiological research", *Emerging Themes in Epidemiology*, vol. 14, no. 1, page 11, 2017, ISSN: 1742-7622. DOI: 10.1186/s12982-017-0064-4. [Online]. Available: https://doi.org/10.1186/s12982-017-0064-4.

[49] *Electric vehicle x driving range prediction github repository*, https://github.com/davidalb97/TFM18-2122i, Accessed: 2022-12-17.

[50] G. S. Oh, David J. Leblanc, and Huei Peng, *Vehicle energy dataset (ved), a large-scale dataset for vehicle energy consumption research*, 2019. arXiv: 1905.02081 [physics.soc-ph].