

A PROGNOSTIC APPROACH TO ALLEVIATE RANGE ANXIETY FOR E-VEHICLES USING ARTIFICIAL INTELLIGENCE

Submitted in partial fulfilment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

KARUMULLA KAUSHIK REDDY (Reg.No - 40110562)

KODALI SAI VENKAT (Reg.No - 40110599)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

CATEGORY-1 UNIVERSITY BY UGC

**Accredited with Grade “A++” by NAAC |12B Status by UGC |Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600119**

APRIL - 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Category - I University by UGC

Accredited "A++" by NAAC | Approved by AICTE

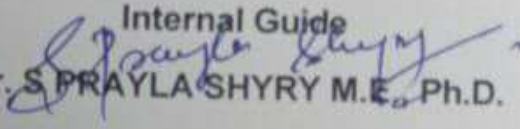
www.sathyabama.ac.in

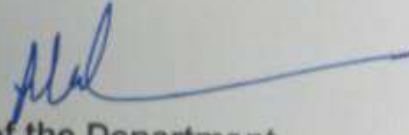
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of KODALI SAI VENKAT(40110599) and KARUMULLA KAUSHIK REDDY(40110562) who carried out the Project entitled "A Prognostic approach to alleviate range anxiety for e - Vehicles using Artificial Intelligence" under my supervision from November 2023 to April 2024.

Internal Guide


Dr. S. PRAYLA SHYRY M.E., Ph.D.


Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.

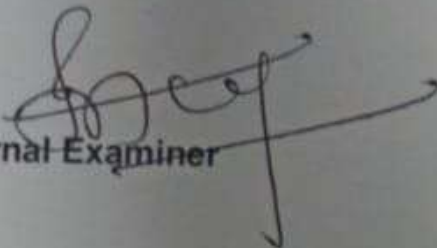
HEAD OF THE DEPARTMENT
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

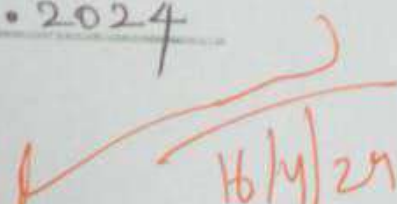
SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Jeppiaar Nagar, Rajiv Gandhi Salai,
Chennai - 600 119

Submitted for Viva voce Examination held on 16.4.2024


Internal Examiner

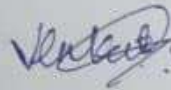

External Examiner

DECLARATION

I, KODALI SAI VENKAT(Reg.No-40110599), hereby declare that the Project Report entitled "A Prognostic approach to alleviate range anxiety for e - Vehicles using Artificial Intelligence." done by me under the guidance of Dr. S. PRAYLA SHYRY, M.E.,Ph.D is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

DATE: 16-04-24

PLACE: Chennai



SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of Management of Sathyabama Institute of Science and Technology** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph. D, Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.S.PRAYLA SHYRY M.E.,Ph.D**,for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Rapid growth in the transportation sector is contributing to pollution. The conventional vehicles which are made of fossil fuels emit harmful gasses that result in global warming. In this scenario e-Vehicles can be chosen as a good alternative but range anxiety is something that is stopping people from resorting to e-Vehicles. Range prediction is a crucial aspect of electric vehicles (EVs) as it directly impacts their usability and practicality. With the increasing adoption of EVs, accurately estimating the range has become essential for optimizing driving plans and addressing range anxiety. Range prediction refers to the estimation of how far an electric vehicle can travel on a single charge based on various factors such as battery capacity, driving conditions, weather, and driving style. It involves the utilization of data-driven models and algorithms that consider these variables to provide an estimate of the remaining range. The development of accurate range prediction models requires a deep understanding EV technology, battery behavior, and real-world driving conditions. By providing drivers with reliable range predictions, they can make informed decisions, plan their journeys more effectively, and enhance their overall experience with electric vehicles. Here proposed machine learning approach that combines physical and environmental factors to accurately predict range and therefore minimize range anxiety. Electric vehicles (EVs) offer a promising solution to mitigate pollution from conventional fossil fuel vehicles, yet range anxiety remains a significant barrier to widespread adoption. Accurate range prediction is pivotal for enhancing the usability and practicality of EVs, particularly as their adoption continues to grow. This prediction involves sophisticated data-driven models considering factors like battery capacity, driving conditions, and weather to estimate the remaining range. Developing precise range prediction models necessitates a comprehensive understanding of EV technology, battery behavior, and real-world driving scenarios. The machine learning approach combines factors to predict electric vehicle range accurately, easing driver anxiety. It optimizes driving plans and empowers EV users with reliable range estimates, the contributing to sustainable mobility.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF TABLES	viii
	LIST OF FIGURES	ix
	LIST OF ABBREVIATIONS	xi
1	INTRODUCTION	1
	1.1 Introduction	1
	1.2 Objectives	2
	1.3 Summary	2
2	LITERATURE SURVEY	3
	2.1 Introduction	3
	2.2 State of Charge	3
	2.3 State of Energy	4
	2.4 State of Health	4
	2.5 Range Anxiety	5
	2.6 Influential Parameters	5
	2.7 Remaining Range Estimation	5
	2.8 Summary	6
3	REQUIREMENT ANALYSIS	10
	3.1 Introduction	10
	3.2 Tools Used	10
	3.3 Libraries Used	10
	3.4 Device Specifications	12
	3.5 Summary	12
4	DESCRIPTION OF PROPOSED SYSTEM	13
	4.1 Introduction	13

	4.2 Work Plan	13
	4.3 Artificial Neural Network	14
	4.4 Long Short-Term Memory	15
	4.5 Architecture Diagram	17
	4.6 Design of Modules	18
	4.7 Summary	18
5	IMPLEMENTATION DETAILS	19
	5.1 Introduction	19
	5.2 Machine Learning Model	19
	5.3 Remaining Range Estimation	21
	5.4 Summary	23
6	RESULTS AND DISCUSSIONS	24
	6.1 Introduction	24
	6.2 Dataset Analysis	24
	6.3 Test Results	31
7	CONCLUSION	35
	7.1 Conclusion	35
	7.2 Future Work	35
	REFERENCES	36
	APPENDIX	38
	(A) SOURCE CODE	38
	(B) SCREEN SHOTS	45
	(C) RESEARCH PAPER	53
	(D) CERTIFICATE	65

LIST OF TABLES

TABLE NO	TITLE	PAGE.NO
2.1	Literature Survey Summary	9
3.1	System Specifications	12
6.1	Comparison Of Test Results	33

LIST OF FIGURES

FIGURE.NO	TITLE	PAGE.NO
4.1	Structure of LSTM	16
4.2	Architecture Diagram	17
5.1	NN for SOC Calculation	20
5.2	NN for Remaining Estimation	22
6.1	City Roads vs Average Trip Distance	24
6.2	Motor Way vs Average Trip Distance	25
6.3	Country Road vs Average Trip Distance	25
6.4	Park Heating vs Average Trip Distance	26
6.5	A/C Vs Average Trip Distance	26
6.6	Trip Distance vs Quantity and Consumption Regression Plot	27
6.7	Trip Distance vs Average Speed Regression Plot	27
6.8	Average Speed Compared with Different Road Types and Driving Styles	28
6.9	Vehicle Consumption Compared with Auxiliary Devices	28
6.10	Histogram for Trip Distance (Target Variable)	29
6.11	Histogram for Quantity	29
6.12	Histogram for Consumption	30
6.13	Histogram for Average Speed	30
6.14	Histogram for ECR Deviation	31
6.15	Mean Absolute Error for Soc Model	31
6.16	Predicted vs Actual SOC Values	31
6.17	Mean Absolute Error for SOE Model	32
6.18	Predicted vs Actual SOE Values	32
6.19	Mean Absolute Error for Range Estimation Model	32
6.20	Predicted vs Actual Range Values	33
6.21	Validation Loss Overtime	34
6.22	MAE Over Time	34
B.1	Importing Libraries and data set	45
B.2	Analyzing the data set	45
B.3	Verifying the variable types	46
B.4	Modifying the dataset	46

B.5	Training the LSTM Model	47
B.6	Checking the validation loss	47
B.7	Validation loss curve	48
B.8	Comparison of actual output and Mean absolute Error	48
B.9	Analyzing the linear Regression Model	49
B.10	Linear Regression Model Comparison	49
B.11	Mean absolute error for Support Vector Regression Model	50
B.12	Support Vector Regression Model Comparison	50
B.13	Mean Absolute error for Random Forest Model	51
B.14	Random Forest Model Comparison	51
B.15	Heatmap Comparison	52

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
AI	Artificial Intelligence
ANN	Artificial Neural Network
EV	Electric Vehicle
LSTM	Long Short-Term Memory
ML	Machine Learning
MLP	Multilayer Perception
NN	Neural Network
RBF	Radial Basis Function
RMS	Root Mean Square
RUL	Remaining Useful Life
RELU	Rectified Linear Unit
SOC	State Of Charge
SOE	State Of Energy
SOH	State Of Health
SOM	Self-Organizing Mapping
SVM	Support Vector Regression

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Transport is a fundamental requirement of modern life, conventional combustion engines that use petrol and diesel are highly polluting and contribute to global warming. These conventional vehicles can be replaced with electric vehicles that have zero tailpipe emissions and are much better for the environment. They have relatively lower running costs, maintenance cost, convenience of charging at home, and are easy to drive and quiet. There is an increase in usage of electrical vehicles lately. It is expected that 145 million electric vehicles will be on the road by 2030. Range prediction is a critical factor for EV users to plan their trips effectively and alleviate range anxiety. Calculating the range of electric vehicles (EVs) using Machine Learning (ML) techniques has gained significant attention in recent years.

ML algorithms leverage historical data from EVs, such as driving patterns, battery characteristics, and external factors like weather and road conditions. By analysing this data, ML libraries can learn complex relationships and patterns that impact the vehicle's range. These LSTM and ANN can then predict the remaining range based on real- time data inputs.

ML-based range prediction systems can continuously adapt and improve their accuracy by learning from a large amount of data collected from various EVs and diverse driving scenarios. These models can account for factors like battery degradation, traffic conditions, driving behavior, and energy consumption patterns, providing more reliable and personalized range estimations. The integration of ML in range prediction for EVs has the potential to enhance the driving experience, optimize route planning, and increase confidence in electric vehicle technology. As research and development in this area advance, ML algorithms can contribute to maximizing the efficiency and usability of EVs, promoting their widespread adoption and contributing to a sustainable transportation future.

1.2 OBJECTIVES

- To design a LSTM model that can accurately predict SOC (State of Charge), SOE (State of Energy) for e-vehicle batteries.
- To estimate and display the remaining coverable range for an electric vehicle using machine learning to reduce range anxiety among users.

1.3 SUMMARY

LSTM and ANN use data like battery info, speed, and temperature to predict how far an electric car can go. By learning from past data and adjusting with real-time info, they get better at estimating range over time. This helps EV drivers plan trips and charging stops more accurately, making their experience smoother and more reliable.

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION

Literature survey explores the various methods proposed for predicting State of charge(SOC), State of energy (SOE) and remaining estimated range.

2.2 STATE OF CHARGE (SOC):

R. Li, H. Sun, X. Wei, W. Ta, and H. Wang, “Lithium Battery State-of-Charge Estimation Based on AdaBoost.Rt-RNN,” *Energies*, vol. 15, no. 16, p. 6056, Aug. 2022. The state of charge (SOC) is a measurement of the amount of energy available in a battery at a specific point in time expressed as a percentage.

$$\text{SOC (\%)} = (Q + Q(\text{initial})) / Q(\text{max}) * 100 \quad \text{Equation 2.1}$$

$Q(\text{initial})$ = Initial charge of the battery.

Q = The quantity of electricity delivered by or supplied to, the battery. It follows the convention of the current: it is negative during the discharge and positive during the charge.

$Q(\text{max})$ = The maximum charge that can be stored in the battery.

R. Li, H. Sun, X. Wei, W. Ta, and H. Wang proposed a method that combines the strengths of AdaBoost.Rt, a boosting algorithm that can improve the accuracy of weak learners, and RNNs, a type of neural network that can model temporal dependencies in time-series data. In this paper [4], the SOC of lithium-ion power batteries was estimated by using a recurrent neural network. The RNN used in this study consists of 6 input neurons, 1 hidden layer and 1 output neuron.

2.3 STATE OF ENERGY (SOE):

O. Rahbari, N. Omar, P. Van Den Bossche and J. Van Mierlo, " Centralizedstate of charge estimation technique for electric vehicles equipped with lithium-ion batteries in smart grid environment," 2018 IEEE International Conference on Industrial Technology (ICIT), Lyon, France, 2018, pp. 1721-1725, doi: 10.1109/ICIT.2018.8352442. The "state of energy" in electric vehicles (EVs) refers to how much electrical energy is stored in the vehicle's battery pack, typically measured as State of Charge (SOC). This information is vital for drivers and vehicle systems as it indicates the available power for driving, helps estimate remaining range, plan charging stops, and optimize energy usage within the vehicle.

2.4 STATE OF HEALTH (SOH):

P. Sharma and B. J. Bora, "A Review of Modern Machine Learning Techniques in the Prediction of Remaining Useful Life of Lithium-Ion Batteries," Batteries, vol. 9, no. 1, p. 13, Dec. 2022. State of health (SOH) is the capability of the battery to retain charge now compared to its rated value. It is defined as the ratio of the maximum battery charge to its rated capacity. It is expressed as a percentage as seen below.

$$\text{SOH (\%)} = (Q(\text{max})/C(\text{rated})) * 100 \quad \text{Equation 2.2}$$

$Q(\text{max})$ = The maximum charge available of the battery.

$C(\text{rated})$ = The rated capacity.

P. Sharma and B. J. Bora employ various machine learning techniques, such as Gaussian process regression and XGBoost, to predict battery Remaining Useful Life (RUL) with high accuracy, achieving less than 5% prediction errors. Meanwhile, Ganesh Sankaran, S. Venkatesan, and M. Prabhakar introduce an LSTM variant with attention mechanisms to better capture complex temporal patterns in battery data. This method improves the relevance of features and enhances information screening for RUL prediction, demonstrated on a dataset of 30 li-ion batteries.

2.5 RANGE ANXIETY:

Miri, I, Fotouhi, A, Ewin, N. Electric vehicle energy consumption modelling and estimation-A case study. *Int J Energy Res.* 2021. Range Anxiety refers to worry on the part of a person driving an e-vehicle that the battery will run out of power before a suitable charging point or destination is reached. Miri, I., Fotouhi, A., and Ewin, N. (2021) explores methods for modeling and estimating energy consumption in electric vehicles, providing valuable insights for optimizing their efficiency and performance. The authors suggest that a range of strategies, including technological advancements, government policies, and public education efforts, could be employed to address this issue and promote the widespread adoption of EVs.

2.6 INFLUENTIAL AUXILIARY PARAMETERS:

Voltage, Current and Temperature form the prime role in estimating the remaining range of an EV. Jun Bi, Yongxing Wang, Qiuyue Sai, Cong Ding, estimating remaining driving range of battery electric vehicles based on real-world data: A case study of Beijing, China, *Energy*, Volume 169, 2019. The authors emphasize the importance of understanding the various factors that influence EV range and highlights the need for accurate range prediction. A Critical Review of Factors considered by S. Sharma, P. D. Achlerkar, P. Shrivastava, A. Garg and B. K. Panigrahi in [12] show that, many factors, viz. vehicle characteristics, driving patterns, environmental conditions & battery aging, battery degradation can impact range over time. Factors such as temperature, charging patterns, and battery chemistry can all contribute to battery degradation.

2.7 REMAINING RANGE ESTIMATION:

Various papers explore electric vehicle range prediction using different methods. Smuts, Scholtz, and Wesson [11] focus on real-time modeling with BMS, GPS, and sensor data. Another study [10] examines factors affecting range prediction, including ML algorithms and environmental conditions. Li et al. [1] and JOUR et al. [9] utilize ML for forecasting based on driving and environmental factors. Szumska and Jurecki [7] integrate physical and data-driven models for accurate prediction.

2.8 Summary

Author	Topic	Method Used	Learnings
Zhigang He a, Xiaoyu Shen a, Yanyan Sun b, Shichao Zhaob, Bin Fan c, Chaofeng Pan	State-of-health estimation and remaining useful life prediction for lithium-ion Battery asedon a ariantlong short term memory neural network	The proposed method is a variant long short term memory neural network (AST-LSTM NN) for state-of-health estimationand remaining useful life prediction of lithium-ion batteries.	AST-LSTM NN effectively predicts multi-step remaining Useful Life (RUL) with its many-to-one structure and active tracking of cell states through element-wise product computation.
Bohan Zheng, Peter He, Lian Zhao, Hongwei Li	A Hybrid Machine Learning Model for Range Estimation of Electric Vehicles	The proposed method is a modified Self-Organizing Maps (SOM) integrating Regression Trees(RT) to predict the power consumption of EV trips.	The Hybrid Model demonstrates superior power consumption estimation for EV trips, revealing EV Big Data characteristics with satisfactory accuracy for both short and long trips in experimental results.
Prabhakar Sharma,Bhaskor J. Bora	A Review of Modern Machine Learning Techniques in the Prediction of Remaining Useful Life of Lithium-Ion Batteries	The investigators have extensively used adaptive neuro-fuzzy inference systems (ANFIS), regression trees (RTs), artificial neural networks (ANN), response surface methodology (RSM) and gene expression programming (GEP).	The paper explores machine learning methods for lithium-ion battery RUL prediction, highlighting their pros, cons, and practical applications in battery management systems.

Ran Li, HuiSun, Xue Wei, Weiwen Ta, Haiying Wang	Lithium Battery State-of-Charge Estimation Based on AdaBoost.Rt-RNN	The paper proposes an integrated learning algorithm of AdaBoost.Rt cyclical neural network model for state-of-charge (SOC) estimation of lithium batteries. The AdaBoost.Rt ensemble learning algorithm is then used to construct a multi-RNN model integration method, with the RNN model as the base learner and AdaBoost.Rt-RNN strong learner model.	The model employs a chain-connected recurrent neural network (RNN) to mitigate neural network algorithm inaccuracies in state-of-charge (SOC) estimation research, promising enhanced accuracy across various complex operating conditions for lithium batteries.
Ganesh Sankaran, S. Venkatesan, M. Prabhakar,	Range Anxiety on electric vehicles in India - Impact on customer and factors influencing range Anxiety	This paper discusses the issue of range anxiety in electric vehicles in India and its impact on customers. It explores the technical factors that contribute to range anxiety and the latest technologies and strategies available to address it.	Strategies to mitigate range anxiety encompass enhancing battery capacity, refining charging infrastructure, deploying smart charging, offering real-time range data, and exploring range extenders like fuel cells or internal combustion engines.

Emilia M. Szumska and Rafał S. Jurecki	Parameters Influencing on Electric Vehicle Range	The authors analyzed data from a real electric vehicle and used big data analysis to estimate the range of an electric vehicle. The big data analysis involved a mathematical model that examined large data sets to discover certain patterns, correlations, or trends that are used to estimate some assumed variable.	The article explains how electric vehicle range is influenced by factors like vehicle design, driving style, and usage conditions, with some factors fixed (e.g., vehicle type, battery weight) and others variable (e.g., battery state, driving behavior).
Jun Bi, Yongxing Wang, Qiuyue Sai, Cong Ding	Estimating remaining driving range of battery electric vehicles based on real-world data: A case study of Beijing, China	The nonlinear estimation models used for remaining driving range under different temperature conditions were established based on the data-driven method. The models consider the State of Charge (SOC), speed and temperature conditions as the impacting factors for remaining driving range	The study stresses considering real-world travel conditions in developing cities like Beijing for accurate estimation of electric vehicle driving range, due to significant variations in traffic and environmental factors compared to developed areas.
JOUR, Sun, Shuai, Zhang,	A Machine Learning	The proposed method is a new driving range	This is a novel method for predicting the driving range of

J. Hong, S. Park and N. Chang	Accurate remaining range estimation for Electric vehicles	The paper suggests a hybrid modeling approach for precise electric vehicle remaining range estimation, beginning with physics-based equations and integrating machine learning to enhance power consumption model accuracy, focusing on refining the first step of a two-step model-based range estimation process.	The paper suggests a hybrid modeling approach to improve electric vehicle range estimation accuracy, combining physics equations with empirical models to potentially increase driving range by up to 30% and boost driver confidence by surpassing current practices.
-------------------------------	---	---	--

Table 2.1 – Literature Survey Summary

Literature survey provides an integral knowledge about the parameters SOC, SOE, SOH, auxiliary parameters that affect the remaining range of an electrical vehicle. The problems in the linear regression and random forest methods are explored and how LSTM helps to overcome this is seen.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 INTRODUCTION

The introduction sets the stage for the experimental methods section, highlighting the use of Python, essential libraries, and the choice of development environments for conducting experiments and analysis.

3.2 TOOLS USED

Visual studio code

Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

Jupyter notebooks

The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser. The Jupyter Notebook App can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet. VS Code supports this natively with help of an extension.

3.3 PROGRAMMING LANGUAGES USED PYTHON

The data preprocessing, machine learning model are built using python with the help of libraries.

3.4 LIBRARIES USED:

3.4.1 Matplotlib

Matplotlib is a powerful Python library for creating 2D plots, offering a range of plot types such as line and bar plots. It's built on NumPy arrays and integrates seamlessly with the SciPy stack, providing visual access to large datasets in digestible formats, making it a popular choice for data visualization tasks.

3.4.2 Sklearn

Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

3.4.3 Tensorflow

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications. TensorFlow is used to create large-scale neural networks with many layers. TensorFlow is mainly used for deep learning or machine learning problems such as Classification, Perception, Understanding, Discovering, Prediction and Creation.

3.4.4 Keras

Keras is a high-level, deep learning API developed by Google for implementing neural networks. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation. TensorFlow is an open-sourced end-to-end platform, a library for multiple machine learning tasks, while Keras is a high-level neural network library that runs on top of TensorFlow.

3.4.5 Numpy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

3.4.6 Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

3.4.7 Seaborn

Seaborn is a library that uses Matplotlib underneath to plot graphs. It will be used to visualize random distributions.

3.4.8 SciPy

SciPy is a Python library for scientific computing, encompassing modules for various tasks such as optimization, linear algebra, signal processing, and more.

3.5 DEVICE SPECIFICATIONS

Processor	Operating system	Memory
1.8 GHz or faster	<i>Windows, linux or macOS</i>	Minimum 8Gb of ram

Table 3.1 – System specifications

3.6 SUMMARY

Requirement analysis employ Python in Visual Studio Code and Jupyter notebooks, utilizing key libraries like Matplotlib, Scikit-learn, TensorFlow, Keras, NumPy, Pandas, Seaborn, and SciPy for data preprocessing and machine learning. Device specifications mandate a processor of 1.8 GHz or faster, Windows, Linux, or macOS operating systems, and a minimum of 8GB of RAM.

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

4.1 INTRODUCTION

The aim is to build a efficient LSTM model that can predict the SOC, SOE and remaining range for an e-vehicle. At first, define the input variables or factors which have greater influence on the end result. This may include battery voltage, current, temperature, vehicle speed, and other sensor readings, as well as data on past charging and discharging cycles. Dataset has to be collected and preprocessed. This data should include a range of operating conditions and should be representative of the target application. The data should also be cleaned, filtered, and normalized to ensure that it is suitable for training the long short term memory model. Using the preprocessed training data, the LSTM model can be trained using a supervised learning approach. Once the LSTM model has been trained, it should be evaluated to ensure that it provides accurate and reliable SOC and SOE estimates under a range of operating conditions. Once the LSTM model has been validated, it can be integrated into the BMS firmware and hardware. Once the ML algorithm has been integrated with the BMS, the entire system should be tested and validated to ensure that it provides accurate and reliable results, and it is important to continuously monitor and update the LSTM model to ensure that it remains accurate and reliable over time.

4.2 WORK PLAN

The module split up is as follows:

Module 1: Collection of data set from e-vehicle using various sensors.

Module 2: State of charge (SOC) calculation by training an ML model.

Module 3: State of energy (SOE) calculation.

Module 4: Training a LSTM model using calculated values of SOC, SOE and other influential parameters to find remaining coverable range.

4.3 ARTIFICIAL NEURAL NETWORK

An Artificial Neural Network (ANN) is a computational model inspired by the structure and functionality of the human brain. It is a powerful tool used in machine learning and artificial intelligence to process and analyze complex data patterns. At its core, an ANN consists of interconnected nodes, called neurons, which mimic the behavior of biological neurons. These neurons work together in layers to perform computations and make predictions. The network receives input data, processes it through the layers of neurons, and produces output predictions or classifications.

The structure of an ANN typically consists of an input layer, one or more hidden layers, and an output layer. The input layer receives the raw data, while the output layer provides the result or prediction. The hidden layers, located between the input and output layers, perform computations and extract features from the data. Each neuron in an ANN receives input signals from the previous layer or directly from the input data. The neuron applies a mathematical transformation to the inputs, known as an activation function, to compute its output. The activation function introduces non-linearity into the network, enabling it to learn complex relationships between inputs and outputs.

The connections between neurons in an ANN are represented by weights. These weights determine the strength of the signal transmitted from one neuron to another. During the learning process, the network adjusts these weights to optimize the model's performance. The weights are initially assigned random values and are updated iteratively using various learning algorithms, such as backpropagation. The learning process in an ANN involves training the network on a labeled dataset. The network compares its predicted output with the desired output and adjusts the weights accordingly. This iterative process continues until the network achieves a satisfactory level of accuracy or minimizes the error. Once trained, the ANN can make predictions or classify new, unseen data based on the patterns it has learned.

Activation functions:

ReLU - The rectifier or ReLU (rectified linear unit) activation function is an activation function defined as the positive part of its argument:

$$f(x) = x^+ = \max(0, x) = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{otherwise.} \end{cases} \quad f'(x) = \begin{cases} 1 & \text{if } x > 0, \\ 0 & \text{if } x < 0. \end{cases}$$

- Equation 4.1

where x is the input to a neuron. This is also known as a ramp function and is analogous to half-wave rectification.

Linear - A linear function is also known as a straight-line function where the activation is proportional to the input i.e., the weighted sum from neurons. It has a simple function with the equation:

$$f(x) = ax + c. \quad \text{-Equation 4.2}$$

4.4 LONG SHORT-TERM MEMORY (LSTM):

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) that is specifically designed to handle sequential data, such as time series, speech, and text. LSTM networks are capable of learning long-term dependencies in sequential data, which makes them well suited for tasks such as language translation, speech recognition, and time series forecasting.

A traditional RNN has a single hidden state that is passed through time, which can make it difficult for the network to learn long-term dependencies. LSTMs address this problem by introducing a memory cell, which is a container that can hold information for an extended period. The memory cell is controlled by three gates: the input gate, the forget gate, and the output gate. These gates decide what information to add to, remove from, and output from the memory cell.

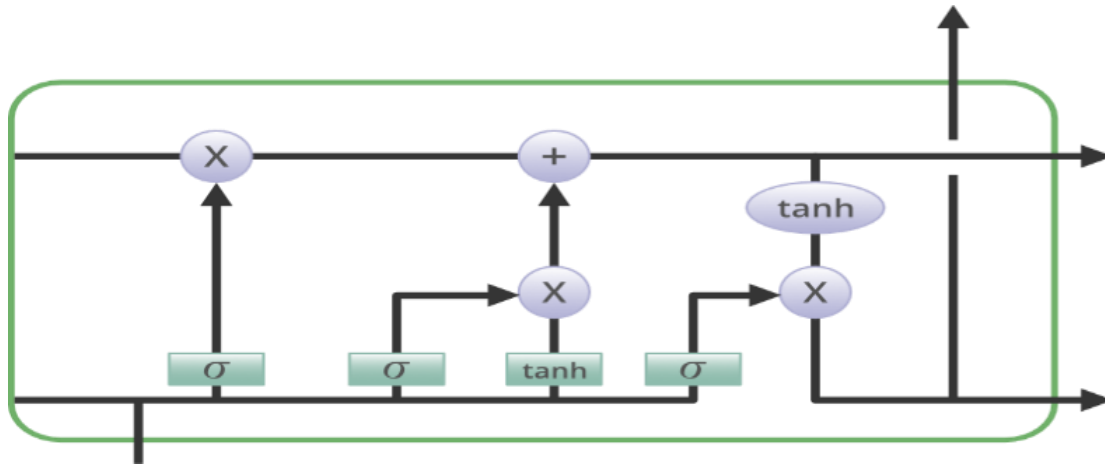


Fig 4.1 - Structure of LSTM

Activation function:

Tanh - The hyperbolic tangent activation function is also referred to simply as the Tanh (also “tanh” and “TanH”) function.

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

- Equation 4.3

The function takes any real value as input and outputs values in the range -1 to 1.

4.5 ARCHITECTURE DIAGRAM:

In fig 4.2, the data collected through Volkswagen automobile company website is preprocessed and divided into testing and training data. An LSTM model is designed to calculate SOC & SOE. Again, another ANN model is built which takes these calculated SOC & SOH values and other additional parameters for training and predicts the remaining range.

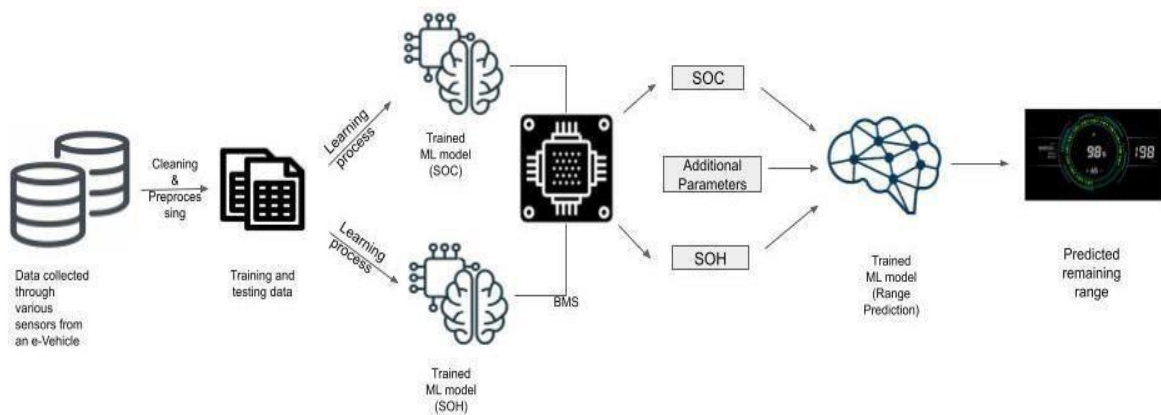


Fig 4.2 - Architecture diagram

The architecture diagram demonstrates the workflow of the project from the pre-processing to delivering output. Architecture diagram is considered as basic outline of the various steps in the project, more over it is like directional map that navigates the order of events in the project.

4.6 DESIGN MODULES

Project Modules:

- Electric Vehicles data extraction
- State of Charge calculation
- State of Energy calculation
- Remaining range estimation

1. Electric Vehicles data extraction

Data such as current, voltage, speed, power, etc. are collected from electric vehicles using various sensors and at various instances. This data is later preprocessed.

2. State of Charge calculation

State of charge refers to the power remaining in the battery. An LSTM and ANN model is trained to calculate the SOC using the attributes current, temperature and voltage.

3. State of Energy calculation

SOE is defined as the ratio of the battery residual energy under specific operating conditions, e.g., varying load and temperature, over the total battery available energy.

4. Remaining range estimation

The remaining range of the vehicle is then predicted/estimated using the calculated SOC, SOE, driving style, road grade, weather conditions, power discharged and auxiliary devices.

4.7 SUMMARY

Aim delves deep into the intricacies of the workflow, elucidating the current range estimation methodologies employed and the intricate web of challenges entwined within. It celebrates the pursuit of knowledge, shedding light on the complex landscape of electric vehicle range prediction, and invites readers to immerse themselves in the journey of discovery and innovation.

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 INTRODUCTION

The implementation involves the combined utilization of Long Short-Term Memory (LSTM) and Artificial Neural Networks (ANN) to enhance the precision of electric vehicle range prediction. Python serves as the programming language, and Google Colab notebooks provide the platform for implementing LSTM and ANN models.

5.2 MACHINE LEARNING MODEL

5.2.1 SOC & SOE:

Data Collection: Drive cycle data from the Panasonic 18650PF Li- ion battery of an EV at normal temperature (25 °C) is used. The drive cycle power profile of an electric Ford F150 truck is collected.

Pre-processing: Data is extracted, outliers removed, and features normalized for LSTM and ANN model training. Only relevant features (Current, Voltage, Battery_temp) are retained to calculate State of Charge (SOC) using a nominal amp-hour (2.5Ah). SOC is derived by adding 2.5Ah to the values and dividing by 2.5Ah, serving as the target variable for the model.

Data Analysis: The dataset doesn't require encoding or imputation due to absence of categorical values and missing data. Pearson correlation coefficient reveals no significant correlations among features. Outliers are detected using Z-Score test and removed. With no issues identified, the dataset is ready for model training.

Data Splitting: 80 percent of data is used for training and 20 percent is for testing.

Model Building:

Code:

```
model_SOC = Sequential()
model_SOC.add(Dense(units=5, activation='relu',
input_dim=len(X_train_SOC.columns)))
model_SOC.add(Dense(units=5, activation='relu'))
model_SOC.add(Dense(units=1, activation='linear'))

es = tf.keras.callbacks.EarlyStopping(monitor='val_loss', mode='min',
verbose=1,patience=50)

opt = keras.optimizers.Adam(0.001)
model_SOC.compile(loss='mse', optimizer=opt,
metrics='mse')model_SOC.summary()
```

Visualization:

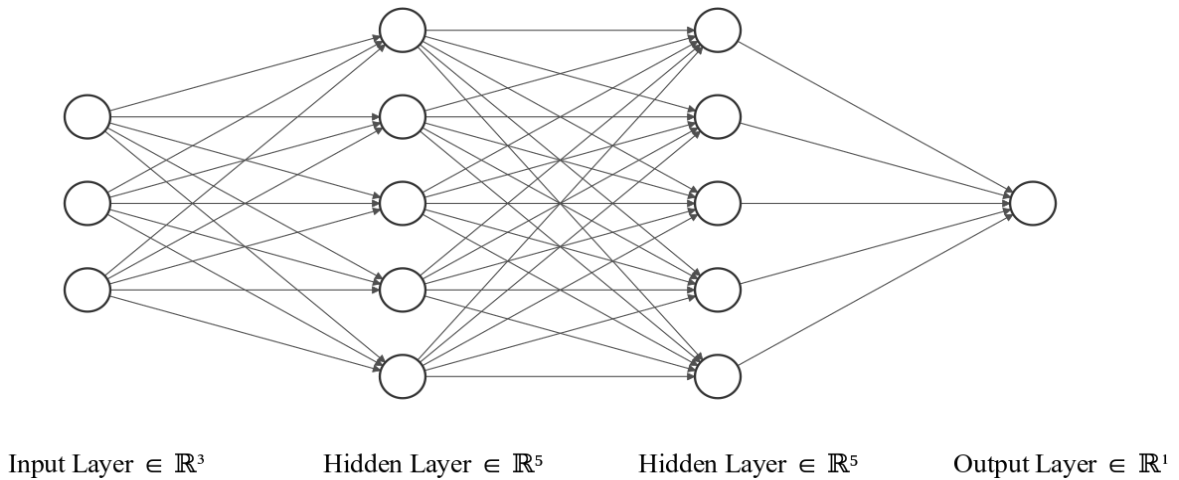


Fig 5.1 – NN for SOC calculation

In the ANN used, the input layer consists of 3 neurons, 2 hidden layers with 5 neurons each and output layer with a single neuron. For hidden layers relu activation function is used and for output layer linear activation function is used. To prevent the model from overfitting early stopping is used and adam optimiser is used.

Training & Testing: Finally, the model is trained and tested.

STATE OF ENERGY (SOE): Data collection and preprocessing involves the same steps as used above, but the SOC as an additional feature is considered.

5.3 REMAINING RANGE ESTIMATION:

Data Collection: For training the LSTM and ANN model, a dataset that contains real-life driving data of two Volkswagen e-Golf cars, with year of manufacture as 2014 and 2016 respectively is used. The file includes data about 3615 trips with a total travel distance of around 152167 kilometers.

Pre-processing: Data preprocessing involved dropping irrelevant features (manufacturer, model, version, power, fuel_date, fuel_type), one-hot encoding categorical variables, and standardizing numerical features. Attributes like trip_distance, quantity, tire_type, city, etc., were retained and converted to compatible data types before model training.

Data Analysis: During analysis, null-value records in the target variable are dropped, while missing values in other attributes are filled with mean or median values. The data is segregated into continuous and discrete features. Continuous variables like trip_distance, quantity, consumption, avg_speed, and ecr_derivation are assessed via histograms to grasp their distributions. Conversely, discrete features such as city, motor_way, country_roads, A/C, and park_heating are examined using graphs to understand their relationships with the target variable.

Encoding: Categorical features such as tyre_type and driving_style are encoded using one-hot encoding.

Outlier analysis: This analysis is performed on continuous features assuming that they are having Gaussian distribution. The data points having a z-score of more than 3 and less than -3 are removed from the dataset since they are considered as outliers.

Correlation b/w features: Based on a heatmap analysis, driving styles Normal and Moderate exhibit a strong negative correlation; Normal is dropped. Similarly, Summer and Winter tire types display a strong negative correlation; Summer is dropped.

Data Splitting: 80 percent of data is used for training and 20 percent is for testing.

Model Building:

Code:

```
model = Sequential()
model.add(LSTM(16,
input_shape=(len(X_train.columns),1),
return_sequences=True))
model.add(LSTM(32,return_sequences=False))
model.add(Dense(units=32, activation='relu'))
model.add(Dense(units=64, activation='relu'))
model.add(Dense(units=1, activation='linear'))
batch_size = 16          # batch size for model fitting
epochs = 1000            # number of epochs for model fitting

STEPS_PER_EPOCH = X_train.shape[0]/batch_size
es = tf.keras.callbacks.EarlyStopping(monitor='val_loss',
patience=50,verbose=1)# Early stopping to prevent overfitting
lr_schedule = tf.keras.optimizers.schedules.InverseTimeDecay(
0.001,
decay_steps=STEPS_PER_EPOCH*100,decay_rate=0.98,
staircase=False)
opt = tf.keras.optimizers.RMSprop(lr_schedule)      above
model.compile(loss='mae', optimizer=opt, metrics=['mae'])
model.summary()
```

Visualization

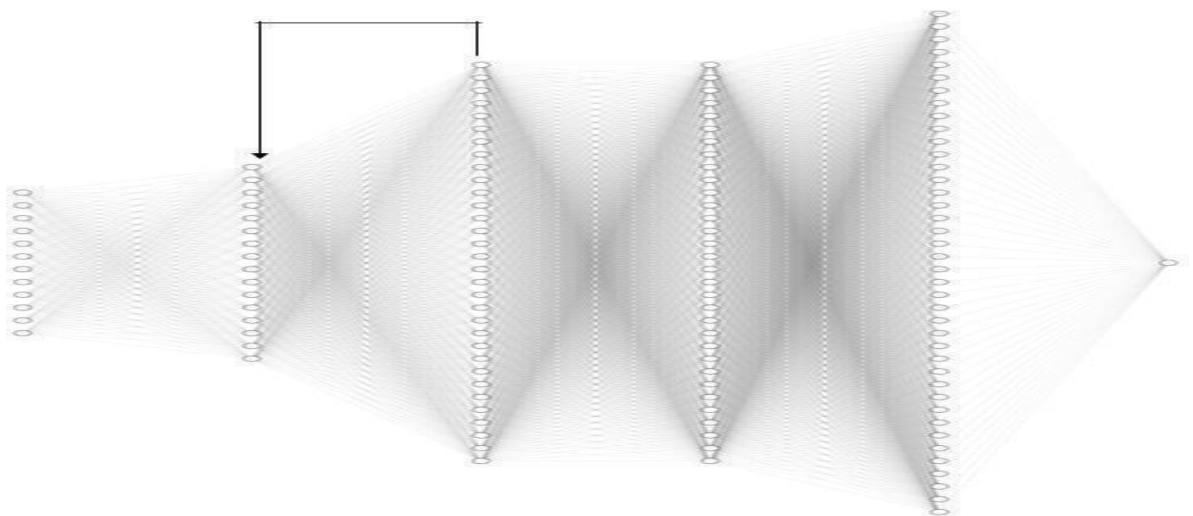


Fig 5.2 – NN for remaining range estimation

In the ANN used, the input layer consists of 12 nodes. There are 4 hidden layers. The first two are LSTM layers (layer1 -16 nodes & layer 2- 32 nodes) and the remaining two(layer 1- 32 nodes & layer 2 -64 nodes) are dense layers. For LSTM layers tanh activation function is used and for dense layers relu is used. The output layer consists of a single node which gives desired output. Linear activation function is used.

Training & Testing: Eighty percent of the collected data is used for training and remaining twenty percent is used for testing the trained LSTM and ANN models performance.

5.4 SUMMARY

SOE and SOC are calculated in one module and are presented as the input to the next SOH module where factors like pack heat, A/C, read type are integrated to find the most accurate range possible.

CHAPTER 6

RESULTS AND DISCUSSION

6.1 INTRODUCTION

People are anxious about range of electric vehicle. A suitable AI model consisting a combination of ANN and LSTM is developed. Thus the problem of range anxiety is addressed.

6.2 DATASET ANALYSIS

Input dataset is analyzed and presented in the pictorial format.

6.2.1 *The trip distance is compared with various parameters*

1) Trip distance compared to city roads

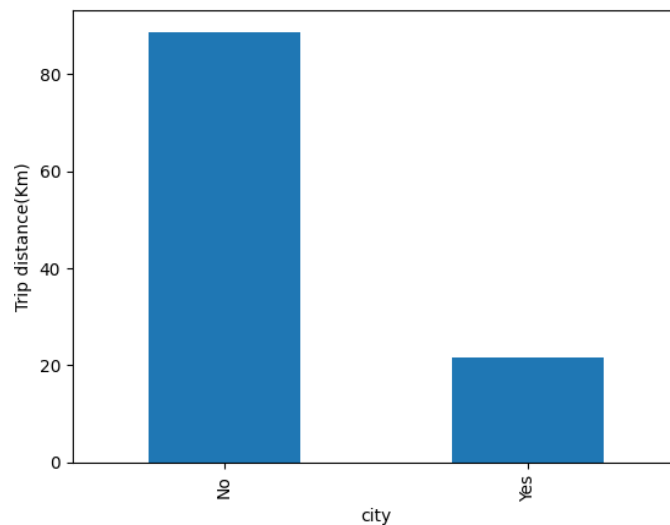


Fig 6.1 – City roads vs average trip distance

Trip distance is lower in cities (This may be due to traffic conditions causes wastage of energy as the brakes are used instead of regenerativebraking.).

2) Trip distance compared to Motor way.

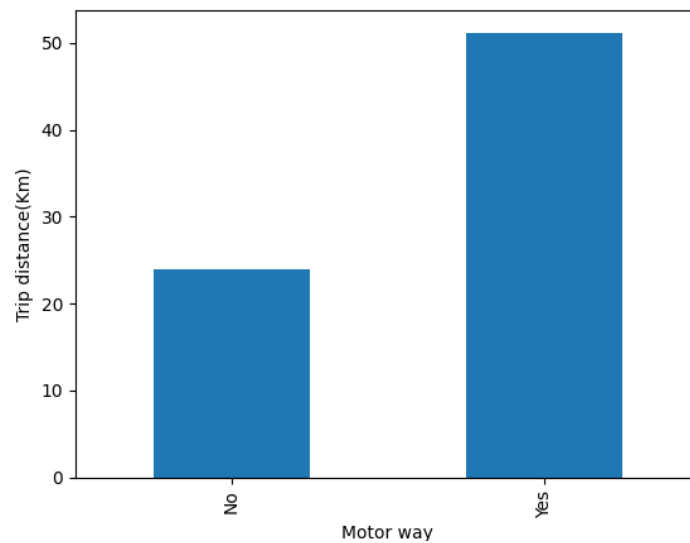


Fig 6.2– motor way vs average trip distance

Trip distance is more in motor ways (This may be because motor ways are free therefore this leads to consistent speeds, reduced stops and reduced accelerations).

3) Trip distance compared to country roads

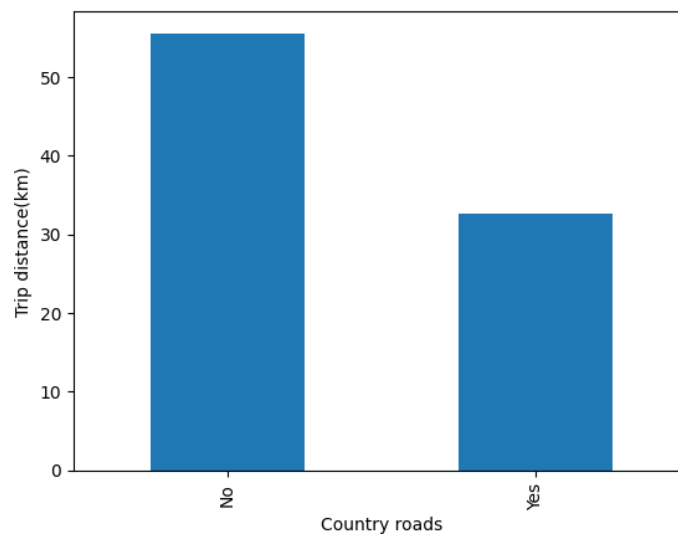


Fig 6.3 – country roads vs average trip distance

Trip distance is less on country roads (This may be because country roads are uneven, have bumps and detours therefore there will be varied speed limits, frequent stops and acceleration and also limited charging infrastructure).

4) Trip distance compared to park heating

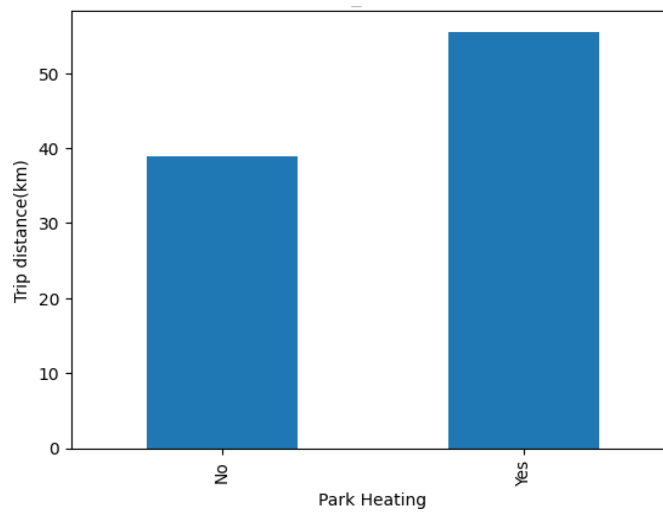


Fig 6.4 – Park heating vs average trip distance

Trip distance is more when park heating is used (park heating can help pre-warm the battery and improve its performance and efficiency).

5) Trip distance compared to air conditioning

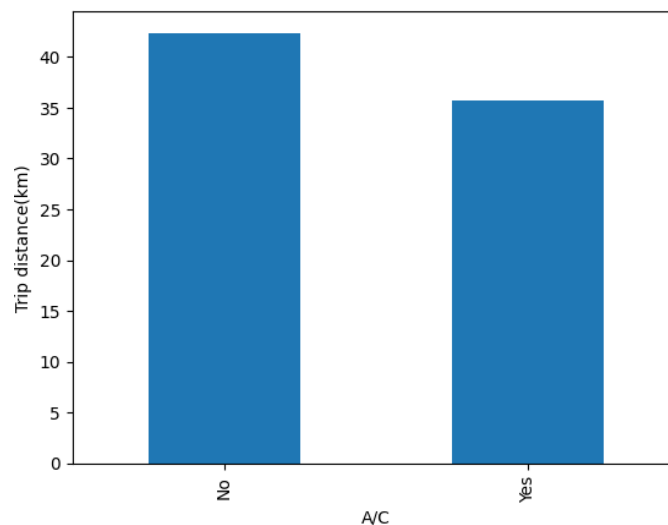


Fig 6.5 – A/C vs average trip distance

Trip distance is less when air conditions is used (air conditioning will result in increased energy consumption and load on the battery).

6) Car battery compared to Trip distance

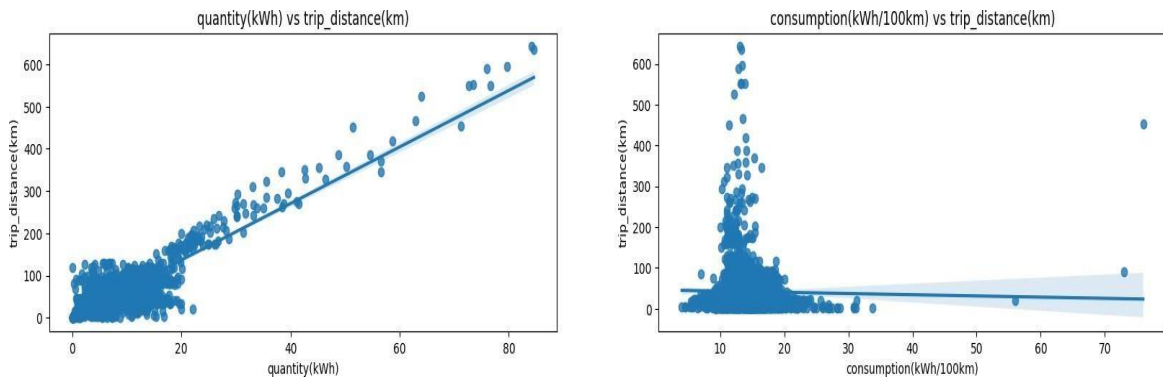


Fig 6.6 – trip distance vs quantity and consumption regression plot

In electric vehicles, trip distance increases with energy quantity above 20 kWh. For energy quantities between 0 to 20 kWh, trip distance depends on additional factors besides energy alone. EVs with energy consumption between 10 to 20 kWh typically cover longer distances per trip.

7) Average speed vs trip distance regression plot

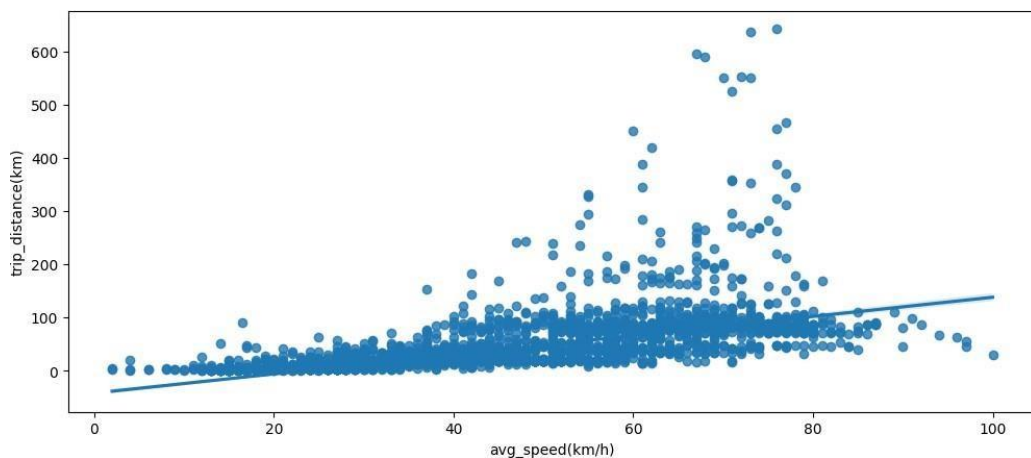


Fig 6.7 – trip distance vs average speed regression plot

We can see data points having an upward trend till 80 km/hr, so the avg_speed will be a feature to consider for distance range.

6.2.2 Average speed distribution

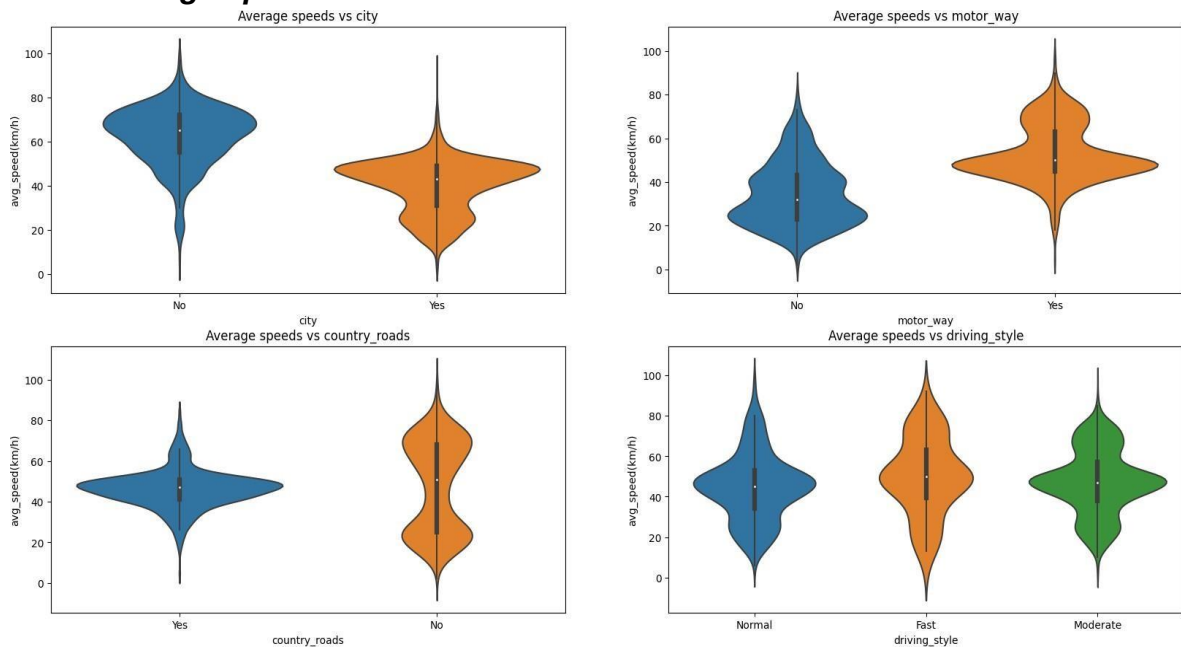


Fig 6.8 – Average speed compared with different road types and driving styles

Inter Quartile Range of Avg_speed differs significantly across cities, with 'No' cities exhibiting higher speeds. Similarly, motorways and country roads with 'Yes' designation tend to have higher average speeds.

6.2.3 Effect of auxiliary devices on consumption

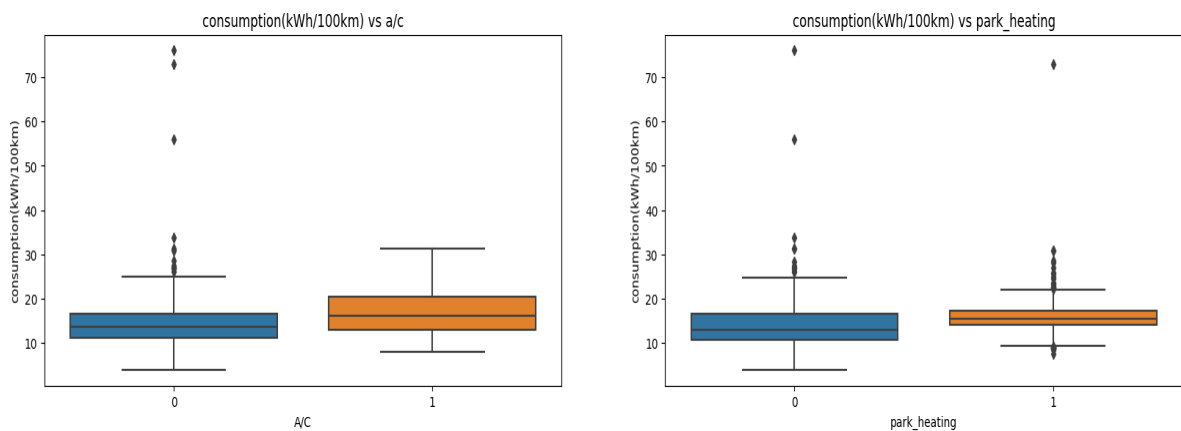


Fig 6.9 – Vehicle consumption compared with auxiliary devices

Energy consumption is comparatively higher when A/C is on and park heat is used because of increased battery usage.

6.2.4 Histograms for continuous variables to visualize their distribution

Histogram is a graphing tool used to summarize discrete or continuous data that are measured on an interval scale.

1) Trip distance

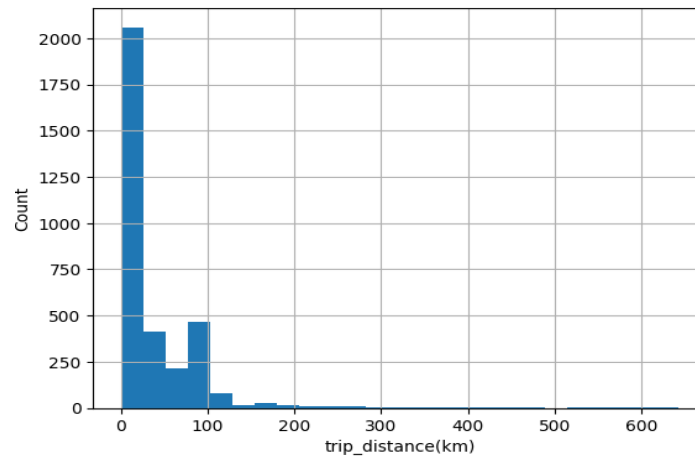


Fig 6.10 - Histogram for trip distance (target variable)

The skewness of the data is positive, indicating that the distribution is skewed towards higher values. The dispersion of trip distance is relatively small, ranging from 1 to 100. Outliers are identified beyond 150 in the trip distance records.

2) Quantity

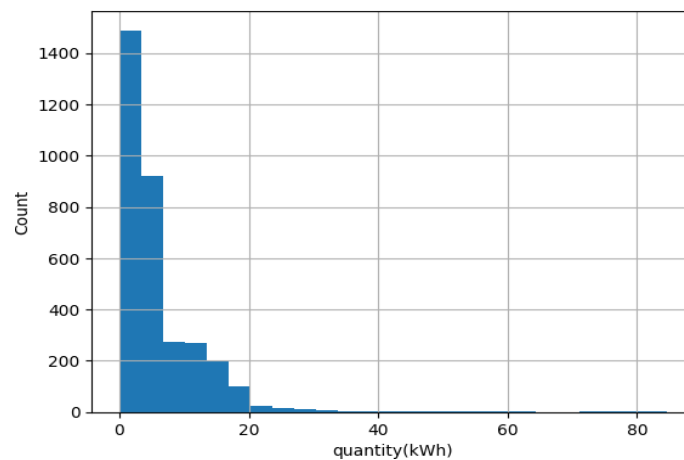


Fig 6.11 - Histogram for quantity

Skewness is positive, indicating a right-skewed distribution. Quantity has a smaller spread, ranging from 0 to 15, with outliers detected in records exceeding 20.

3) Consumption

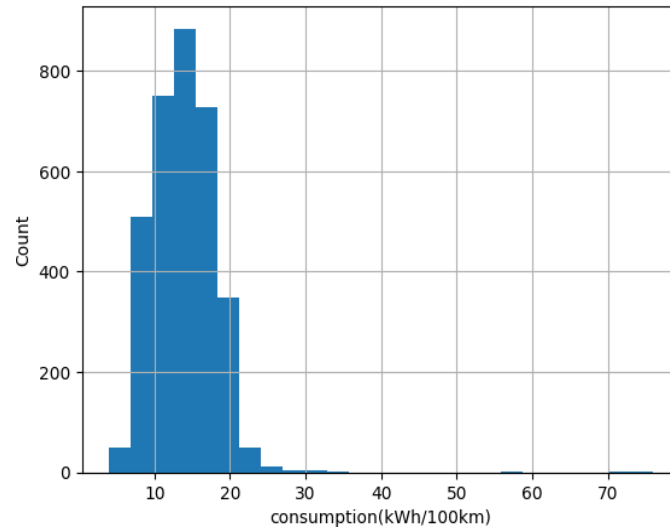


Fig 6.12 - Histogram for consumption

Consumption data shows positive skewness, indicating a right-leaning distribution. It typically ranges from 5 to 20 units, with outliers observed records surpassing 25.

4) Average speed

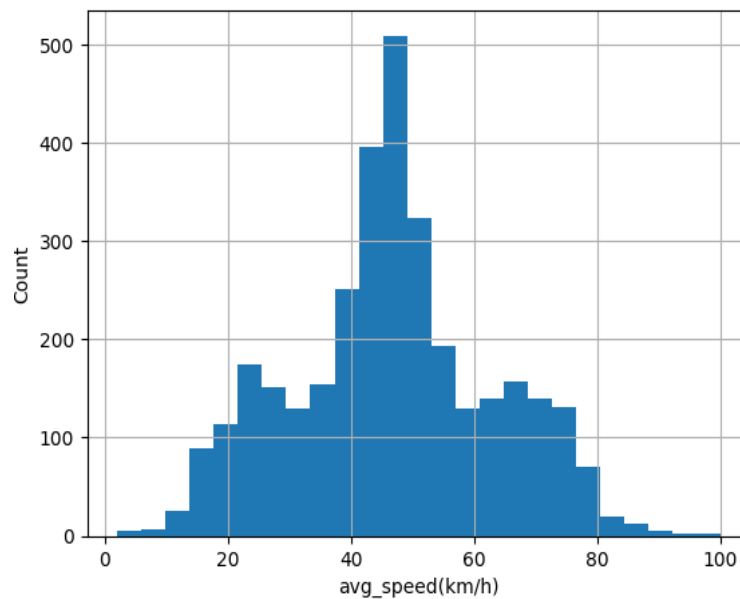


Fig 6.13 - Histogram for average speed

The data exhibits no skewness, with an evenly spread dispersion between 0 to 15, except for outliers beyond 80.

5) ECR deviation

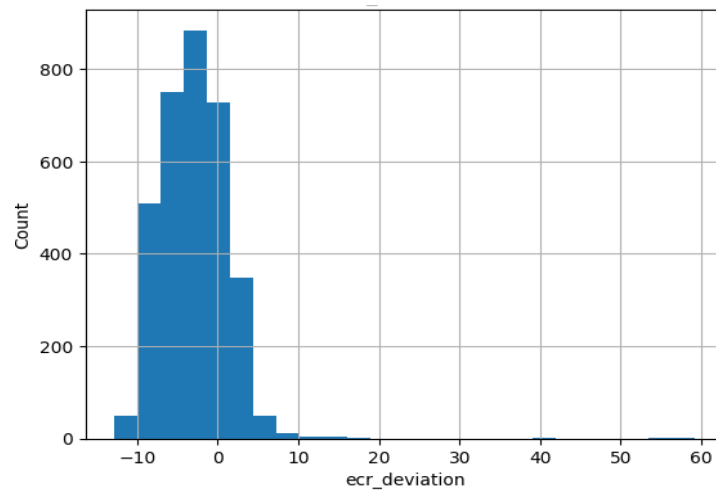


Fig 6.14 - Histogram for ecr deviation

ECR deviation shows positive skewness, concentrated around -10 to 5, with outliers observed in records exceeding 5.

6.3 TEST RESULTS

6.3.1 SOC

Error:

```
641/641 [=====] - 1s 763us/step
Mean absolute error: 0.014113650812155867
```

Fig 6.15 – Mean absolute error for SOC model

Test: Prediction vs Actual values:

	Predictions	Actual Output
0	0.135118	0.167040
1	0.793014	0.791524
2	0.249730	0.254800
3	0.279199	0.282960
4	0.816404	0.790392
...	...	...
20484	0.147491	0.162160
20485	0.984723	0.928672
20486	0.316771	0.315720
20487	0.878058	0.884896
20488	0.284465	0.284760

20489 rows × 2 columns

Fig 6.16 – Predicted vs actual SOC values

6.3.2 SOE

Error:

```
641/641 [=====] - 0s 600us/step
Mean absolute error: 0.0008240243712780419
```

Fig 6.17 – Mean absolute error for SOE model

Test: prediction vs Actual values:

	Predictions	Actual Output
0	0.161298	0.161857
1	0.777971	0.777673
2	0.244460	0.244004
3	0.271357	0.271443
4	0.777336	0.776499
...	...	...
20484	0.157434	0.157908
20485	0.920528	0.921808
20486	0.302690	0.302908
20487	0.874473	0.874161
20488	0.272930	0.273210

20489 rows × 2 columns

Fig 6.18 – Predicted vs actual SOE values

6.3.3 Range Estimation:

Error:

```
from sklearn.metrics import mean_absolute_error
predictions = (model.predict(X_test))
Y_test.to_csv('./y_test.csv', index= False)
pd.DataFrame(predictions).to_csv('./predictions.csv', index= False)
print(mean_absolute_error(Y_test, predictions))
✓ 2.4s

21/21 [=====] - 2s 6ms/step
5.967760939558764
```

Fig 6.19 – Mean absolute error for Range estimation model

Test: Prediction vs Actual values:

	Predictions	Actual Output
0	19.913588	20.0
1	19.177187	19.0
2	20.933769	22.0
3	20.233088	20.0
4	19.742170	20.0
...	...	...
662	142.805740	143.0
663	23.974070	12.0
664	4.279182	5.0
665	20.014788	20.0
666	31.427786	33.0
667 rows × 2 columns		

Fig 6.20 – Predicted vs actual range values

From the comparison table, the margin of errors is observed to be minimal when compared to SVR.

Comparison with other models:

Model	MAE Value	Parameters
SVR	15.25	Kernel = rbf
Linear regression	10.28	Multi valued
Random forest	6.78	n_estimators = 12
MLP	5.96	batchsize = 16 optimizer = RMSprop callbacks = earlystopping

Table 6.1– Comparison of test results

From the above table it can be observed that the mean absolute error is least for MLP. So, MLP is most reliable for remaining range prediction.

Validation loss vs epochs:

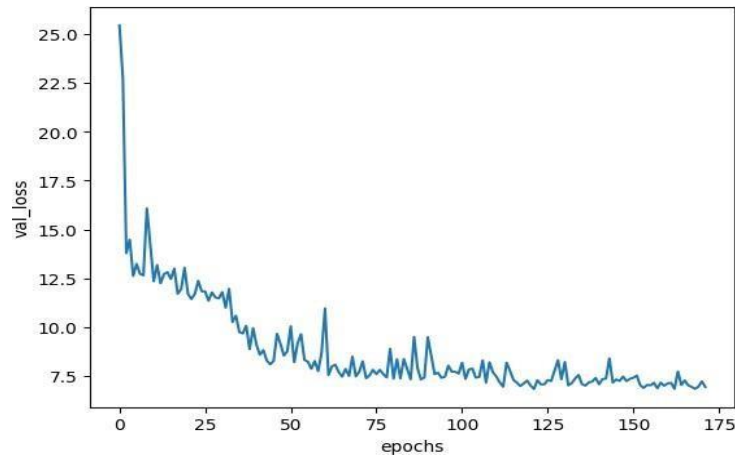


Fig 6.21 – Validation loss over time

The validation loss started at 25.0 and progressively decreased to 7 after 175 epochs, indicating improved model performance. This reduction demonstrates the efficacy of the training process in minimizing prediction errors.

Mean absolute error over time:

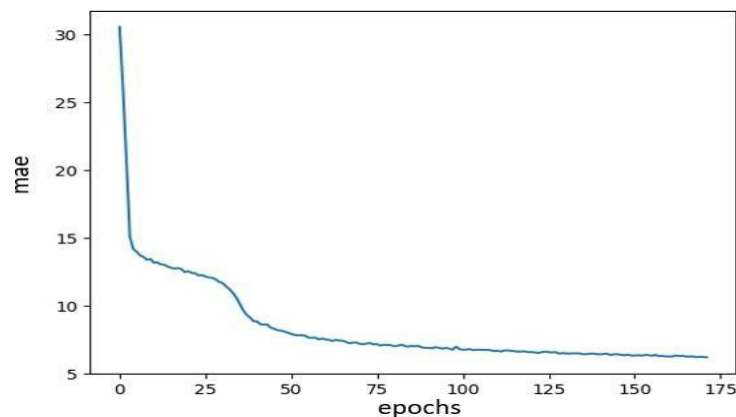


Fig 6.22 – MAE over time

The Mean Absolute Error (MAE) initially stood at 30.0, and it decreased to 5 after 175 epochs, indicating substantial improvement in prediction accuracy during model training. This reduction underscores the model's enhanced performance over successive iterations.

CHAPTER 7

CONCLUSION

7.1 CONCLUSION:

Artificial Intelligence is utilized to implement a prognostic approach, effectively mitigating range anxiety in electric vehicles. The application of the machine learning libraries for calculating the State of Charge (SOC) has yielded highly promising results, showcasing its effectiveness in accurately predicting the State of Energy (SOE) by leveraging updated SOC values. The LSTM and ANN performance in estimating the remaining range has been particularly noteworthy, with the Mean Absolute Error (MAE) recorded at 5.96. This MAE value signifies a close alignment between the predicted and actual outputs, indicating the model's ability to provide reliable estimations of remaining range. These findings not only highlight the LSTM and ANN robustness but also establish its practical suitability, suggesting its potential application in real-world scenarios. Such accuracy and reliability are pivotal for electric vehicle (EV) users and manufacturers alike, as they enable informed decision-making regarding charging schedules, driving routes, and overall energy management strategies. Consequently, the demonstrated efficacy of this ML-based approach holds significant implications for enhancing the efficiency and usability of EVs in everyday contexts.

7.2 FUTURE WORK:

The proposed model focuses on predicting the remaining range of an electric vehicle (e-vehicle). It considers various parameters including road grade, driver behavior, environmental factors, and auxiliary devices. Additionally, the model recognizes that seating capacity can impact the range and can be incorporated as a constraint for future retraining. This allows the model to improve its accuracy in predicting the range over time by considering these additional factors.

REFERENCES

- [1] Penghua Li, Zijian Zhang, Qingyu Xiong, Baocang Ding, Jie Hou, Dechao Luo, Yujun Rong, Shuaiyong Li, "State-of-health estimation and remaining useful life prediction for the lithium-ion battery based on a variant long short term memory neural network", *Journal of Power Sources*, Volume 459, 2020.
- [2] B. Zheng, P. He, L. Zhao and H. Li, "A Hybrid Machine Learning Model for Range Estimation of Electric Vehicles," 2016 IEEE Global Communications Conference (GLOBECOM), Washington, DC, USA, 2016, pp. 1-6, doi: 10.1109/GLOCOM.2016.7841506.
- [3] P. Sharma and B. J. Bora, "A Review of Modern Machine Learning Techniques in the Prediction of Remaining Useful Life of Lithium-Ion Batteries," *Batteries*, vol. 9, no. 1, p. 13, Dec. 2022.
- [4] R. Li, H. Sun, X. Wei, W. Ta, and H. Wang, "Lithium Battery State-of-Charge Estimation Based on AdaBoost.Rt-RNN," *Energies*, vol. 15, no. 16, p. 6056, Aug. 2022.
- [5] Ganesh Sankaran, S. Venkatesan, M. Prabhakar, Range Anxiety on electric vehicles in India -Impact on customer and factors influencing range Anxiety, *Materials Today: Proceedings*, Volume 33, Part 1, 2020.
- [6] Miri, I, Fotouhi, A, Ewin, N. Electric vehicle energy consumption modelling and estimation—A case study. *Int J Energy Res*. 2021.
- [7] Szumska, E.M.; Jurecki, R.S. Parameters Influencing on Electric Vehicle Range. *Energies* 2021, 14, 4821.
- [8] Jun Bi, Yongxing Wang, Qiuyue Sai, Cong Ding, estimating remaining driving range of battery electric vehicles based on real-world data: A case study of Beijing, China, *Energy*, Volume 169, 2019, Pages 833-843, ISSN 0360-5442.
- [9] JOUR, Sun, Shuai, Zhang, Jun, Bi, Jun, Wang, Yongxing, 2019/01/09, A Machine Learning Method for Predicting Driving Range of Battery Electric Vehicles, 4109148.
- [10] J. Hong, S. Park and N. Chang, "Accurate remaining range estimation for Electric vehicles," 2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC), Macao, China, 2016.

- [11] M. Smuts, B. Scholtz and J. Wesson, "A critical review of factors influencing the remaining driving range of electric vehicles," 2017 1st International Conference on Next Generation Computing Applications (NextComp), Mauritius, 2017.
- [12] S. Sharma, P. D. Achlerkar, P. Shrivastava, A. Garg and B. K. Panigrahi, "Combined SoC and SoE Estimation of Lithium-ion Battery using Multi-layer Feedforward Neural Network," 2022 IEEE International Conference on Power Electronics, Drives and Energy Systems (PEDES), Jaipur, India, 2022, pp. 1-6, doi: 10.1109/PEDES56012.2022.10080110.
- [13] O. Rahbari, N. Omar, P. Van Den Bossche and J. Van Mierlo, "A centralized state of charge estimation technique for electric vehicles equipped with lithium-ion batteries in smart grid environment," 2018 IEEE International Conference on Industrial Technology (ICIT), Lyon, France, 2018, pp. 1721-1725, doi: 10.1109/ICIT.2018.8352442.
- [14] R. Zhi-yong, "Research on influence factors affecting driving range of flame-proof battery electric vehicles," 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chengdu, China, 2019, pp. 1982-1986, doi: 10.1109/IAEAC47372.2019.8997686.

APPENDIX

A. SOURCE CODE

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

## Display all the columns of the dataframe
pd.pandas.set_option('display.max_columns',None)
data=pd.read_csv('./volkswagen_e_golf.csv')
## print shape of dataset with rows and columns
print(data.shape)
##type conversion
def tryconvert(value, default, t):
    if t == 'numeric':
        try:
            return float(value)
        except (ValueError, TypeError):
            return default
    if t == 'str':
        try:
            return str(value)
        except (ValueError, TypeError):
            return default
##type conversion
def tryconvert(value, default, t):
    if t == 'numeric':
        try:
            return float(value)
        except (ValueError, TypeError):
            return default
    if t == 'str':
```

```

    try:
        return str(value)
    except (ValueError, TypeError):
        return default
for feature in continuous_feature:
    data2=data.copy()
    data2[feature].hist(bins=25)
    plt.xlabel(feature)
    plt.ylabel("Count")
    plt.title(feature)
    plt.show()

plt.figure(figsize=(20, 10))
plt.subplot(2, 2, 1)
plt.title("Average speeds vs city")
sns.violinplot(x = 'city', y = 'avg_speed(km/h)', data = data)
plt.subplot(2, 2, 2)
plt.title("Average speeds vs motor_way")
sns.violinplot(x = 'motor_way', y = 'avg_speed(km/h)', data = data)
plt.subplot(2, 2, 3)
plt.title("Average speeds vs country_roads")
sns.violinplot(x = 'country_roads', y = 'avg_speed(km/h)', data = data)
plt.subplot(2, 2, 4)
plt.title("Average speeds vs driving_style")
sns.violinplot(x = 'driving_style', y = 'avg_speed(km/h)', data = data)
plt.show()

## role of avg_speed in driving_range
plt.figure(figsize=(12, 5))
plt.title("average speed vs driving range")
sns.regplot(x = 'avg_speed(km/h)', y = 'trip_distance(km)', data = data)
plt.show()

## Quantity on trip distance

```

```

plt.figure(figsize=(20, 8))
plt.suptitle("car battery info. vs driving range")
plt.subplot(2, 2, 1)
plt.title("quantity(kWh) vs trip_distance(km)")
sns.scatterplot(x = 'quantity(kWh)', y = 'trip_distance(km)', data = data)

plt.subplot(2, 2, 2)
plt.title("consumption(kWh/100km) vs trip_distance(km)")
sns.scatterplot(x = 'consumption(kWh/100km)', y = 'trip_distance(km)', data = data)
plt.show()
### a/c and park_heating effect on energy consumption

plt.figure(figsize=(20, 10))
plt.subplot(2, 2, 1)
plt.title("consumption(kWh/100km) vs a/c")
sns.boxplot(x = 'A/C', y = 'consumption(kWh/100km)', data = data)

plt.subplot(2, 2, 2)
plt.title("consumption(kWh/100km) vs park_heating")
sns.boxplot(x = 'park_heating', y = 'consumption(kWh/100km)', data = data)
### a/c and park_heating effect on energy consumption

plt.figure(figsize=(20, 10))
plt.subplot(2, 2, 1)
plt.title("consumption(kWh/100km) vs a/c")
sns.boxplot(x = 'A/C', y = 'consumption(kWh/100km)', data = data)

plt.subplot(2, 2, 2)
plt.title("consumption(kWh/100km) vs park_heating")
sns.boxplot(x = 'park_heating', y = 'consumption(kWh/100km)', data = data)

from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy='mean')

```

```

cols_with_missing_data = [cols for cols in X_train.columns
                           if X_train[cols].isnull().any()]

X_train_imputed_data_columns =
pd.DataFrame(imputer.fit_transform(X_train[cols_with_missing_data]))
X_test_imputed_data_columns =
pd.DataFrame(imputer.transform(X_test[cols_with_missing_data]))

X_train_imputed_data_columns.columns = cols_with_missing_data
X_test_imputed_data_columns.columns = cols_with_missing_data

X_train = pd.concat([X_train.drop(cols_with_missing_data, axis=1, inplace=False),
X_train_imputed_data_columns], axis=1)
X_test = pd.concat([X_test.drop(cols_with_missing_data, axis=1, inplace=False),
X_test_imputed_data_columns], axis=1)

plt.figure(figsize=(6,6))
plt.hist(X_train['avg_speed(km/h)'], bins=30, rwidth=0.8)
plt.xlabel('avg_speed(km/h)')
plt.ylabel('Count')
plt.show()
import numpy as np
from scipy.stats import norm
%matplotlib inline

plt.figure(figsize=(6,6))
rng = np.arange(X_train['avg_speed(km/h)'].min(),
X_train['avg_speed(km/h)'].max(), 0.1)
plt.plot(rng, norm.pdf(rng, X_train['avg_speed(km/h)'].mean(),
X_train['avg_speed(km/h)'].std()))
plt.show()

plt.boxplot(X_train['avg_speed(km/h)'])
plt.ylabel('avg_speed(km/h)')

```

```
plt.title('avg_speed(km/h)')
plt.show()
```

```
plt.figure(figsize=(6,6))
plt.hist(X_train['quantity(kWh)'], bins=30, rwidth=0.8)
plt.xlabel('quantity(kWh)')
plt.ylabel('Count')
plt.show()
```

```
plt.figure(figsize=(6,6))
rng = np.arange(X_train['quantity(kWh)'].min(), X_train['quantity(kWh)'].max(), 0.1)
plt.plot(rng, norm.pdf(rng, X_train['quantity(kWh)'].mean(),
X_train['quantity(kWh)'].std()))
plt.show()
```

```
plt.boxplot(X_train['quantity(kWh)'])
plt.ylabel('quantity(kWh)')
plt.title('quantity(kWh)')
plt.show()
```

```
to_be_included_train = X_train['quantity(kWh)']<=65
to_be_included_test = X_test['quantity(kWh)']<=65
```

```
X_train = X_train[to_be_included_train]
Y_train = Y_train[to_be_included_train]
```

```
X_test = X_test[to_be_included_test]
Y_test = Y_test[to_be_included_test]
```

```
print(X_train.shape, Y_train.shape, X_test.shape, Y_test.shape)
```

```
X_train.reset_index(level=None, drop=True, inplace=True, col_level=0, col_fill="")
X_test.reset_index(level=None, drop=True, inplace=True, col_level=0, col_fill="")
Y_train.reset_index(level=None, drop=True, inplace=True)
```

```

Y_test.reset_index(level=None, drop=True, inplace=True)
cols_to_be_normalized = ['quantity(kWh)', 'avg_speed(km/h)']
X_train = X_train.drop(['driving_style Normal', 'tire_type Summer tires'], axis=1)
X_test = X_test.drop(['driving_style Normal', 'tire_type Summer tires'], axis=1)

for each in cols_to_be_normalized:
    mean = X_train[each].mean()
    std = X_train[each].std()

    X_train[each] = (X_train[each] - mean)/std
    X_test[each] = (X_test[each] - mean)/std

model = Sequential()
model.add(LSTM(16, input_shape=(len(X_train.columns),1),
return_sequences=True))
model.add(LSTM(32,return_sequences=False))
# model.add(Dense(units=32, activation='relu',input_dim=len(X_train.columns)))
model.add(Dense(units=32, activation='relu'))
model.add(Dense(units=64, activation='relu'))
# model.add(Dense(units=4, activation='relu'))
model.add(Dense(units=1, activation='linear'))

batch_size = 16    # batch size for model fitting

epochs = 1000    # number of epochs for model fitting

STEPS_PER_EPOCH = X_train.shape[0]/batch_size

es = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=50,verbose=1)
# Early stopping to prevent overfitting

lr_schedule = tf.keras.optimizers.schedules.InverseTimeDecay(          # Using
a dynamic learning rate that decays a certain rate per 100 epochs

```

```

0.001,
decay_steps=STEPS_PER_EPOCH*100,
decay_rate=0.98,
staircase=False)

```

```

opt = tf.keras.optimizers.RMSprop(lr_schedule) # Using RMSprop optimizer with
the learning rate schedule as mentioned above

```

```

model.compile(loss='mae', optimizer=opt, metrics=['mae'])
model.summary()
plt.ylabel('val_loss')
plt.xlabel('epochs')
plt.plot(h['val_loss'])
plt.show()
plt.ylabel('loss')
plt.xlabel('epochs')
plt.plot(h['loss'])
plt.show()
plt.ylabel('mae')
plt.xlabel('epochs')
plt.plot(h['mae'])
import pickle
pickle.dump(model,open('range_model.pkl','wb'))
from sklearn.metrics import mean_absolute_error
predictions = (model.predict(X_test))
Y_test.to_csv('./y_test.csv', index= False)
pd.DataFrame(predictions).to_csv('./predictions.csv', index= False)
print(mean_absolute_error(Y_test, predictions))
compare = pd.concat([pd.DataFrame(predictions), pd.DataFrame(Y_test)], axis=1)
compare.columns = ['Predictions', 'Actual Output']
compare.head()

```

B.SCREENSHOTS

DATA IN

```
## Data Analysis Phase
## Main aim is to understand more about the data

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
## Display all the columns of the dataframe
pd.pandas.set_option('display.max_columns',None)

[ ] data=pd.read_csv('/content/volkswagen_e_golf.csv')

## print shape of dataset with rows and columns
print(data.shape)
```

Fig. B.1 Importing Libraries and dataset

(3345, 18)

```
# print the top5 records
data.head()
```

	manufacturer	model	version	power(kW)	fuel_date	trip_distance(km)	quantity(kWh)	fuel_type	tire_type	city	motor_way	country_roads
0	Volkswagen	Golf	e-Golf	85	08.02.2019	80	NaN	Electricity	Winter tires	0	0	1
1	Volkswagen	Golf	e-Golf	85	06.02.2019	50	12.29	Electricity	Winter tires	0	0	1
2	Volkswagen	Golf	e-Golf	85	05.02.2019	43	8.68	Electricity	Winter tires	0	1	1
3	Volkswagen	Golf	e-Golf	85	04.02.2019	44	1.50	Electricity	Winter tires	0	1	1
4	Volkswagen	Golf	e-Golf	85	04.02.2019	76	14.44	Electricity	Winter tires	0	1	0

Fig. B.2 Analyzing the data set

```

▶ #type checking
data.dtypes

➡ manufacturer      object
model               object
version             object
power(kW)           int64
fuel_date           object
trip_distance(km)   object
quantity(kWh)       float64
fuel_type           object
tire_type           object
city               int64
motor_way           int64
country_roads       int64
driving_style       object
consumption(kWh/100km) float64
A/C                int64
park_heating        int64
avg_speed(km/h)     float64
ecr_deviation       float64
dtype: object

```

Fig. B.3 Verifying the variable types

```

▶ #type conversion
def tryconvert(value, default, t):
    if t == 'numeric':
        try:
            return float(value)
        except (ValueError, TypeError):
            return default
    if t == 'str':
        try:
            return str(value)
        except (ValueError, TypeError):
            return default

[ ]
#trip distance is a float value
data['trip_distance(km)'] = data['trip_distance(km)'].map(lambda x : tryconvert(x, np.nan, 'numeric') )

```

Fig. B.4 Modifying the dataset

- Training the model

```

hs = model.fit(X_train, Y_train, epochs=epochs, batch_size=batch_size, callbacks=[es], validation_split=0.2, v
...
134/134 [=====] - 1s 7ms/step - loss: 5.0906 - mae: 5.0906 - val_loss: 5.9330 - val_
Epoch 305/1000
134/134 [=====] - 1s 7ms/step - loss: 5.0647 - mae: 5.0647 - val_loss: 5.8766 - val_
Epoch 306/1000
134/134 [=====] - 1s 7ms/step - loss: 5.0691 - mae: 5.0691 - val_loss: 5.8618 - val_
Epoch 307/1000
134/134 [=====] - 1s 7ms/step - loss: 5.0699 - mae: 5.0699 - val_loss: 5.9303 - val_
Epoch 308/1000
134/134 [=====] - 1s 9ms/step - loss: 5.0859 - mae: 5.0859 - val_loss: 5.9015 - val_
Epoch 309/1000
134/134 [=====] - 1s 9ms/step - loss: 5.0541 - mae: 5.0541 - val_loss: 5.9469 - val_
Epoch 310/1000
134/134 [=====] - 1s 7ms/step - loss: 5.0632 - mae: 5.0632 - val_loss: 5.9929 - val_
Epoch 311/1000
134/134 [=====] - 1s 7ms/step - loss: 5.0384 - mae: 5.0384 - val_loss: 5.8710 - val_
) > fit() > error_handler() > _call__() > _call() > call_function() > _call_flat() > call_prepared() > call_flat() > call_function() > quick_execute() ... >

```

Fig B.5 Training the LSTM model

```

▶ plt.ylabel('val_loss')
  plt.xlabel('epochs')
  plt.plot(h['val_loss'])
  plt.show()

```

Fig B.6 Checking the validation loss

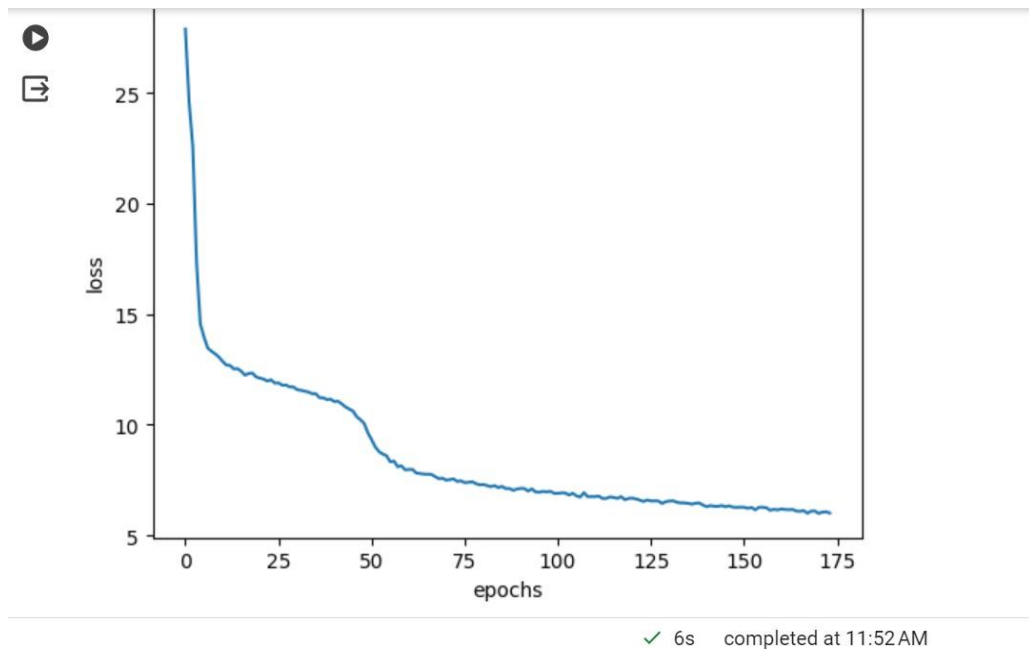


Fig B.7 Validation loss curve

```

✓ 0s ▶ from sklearn.metrics import mean_absolute_error
      predictions = (model.predict(X_test))
      Y_test.to_csv('./y_test.csv', index= False)
      pd.DataFrame(predictions).to_csv('./predictions.csv', index= False)
      print(mean_absolute_error(Y_test, predictions))

21/21 [=====] - 1s 3ms/step
6.919671273517038

✓ 0s [58] compare = pd.concat([pd.DataFrame(predictions), pd.DataFrame(Y_test)], axis=1)
      compare.columns = ['Predictions', 'Actual Output']
      compare.head()

```

	Predictions	Actual Output
0	6.675969	7.0
1	52.306446	39.0
2	2.175501	2.0
3	46.418659	52.0

Fig B.8 Comparison of actual output and Mean absolute Error

```

from sklearn import linear_model
reg = linear_model.LinearRegression()

# train the model using the training sets
reg.fit(X_train, Y_train)

# regression coefficients
print('Coefficients: ', reg.coef_)

# variance score: 1 means perfect prediction
print('Variance score: {}'.format(reg.score(X_test, Y_test)))

```

Coefficients: [-6.56257482 -1.93379935 -0.19431879 -0.56875174 1.7
 -10.44913083 -0.56875174 -1.5068202 -1.80629492 0.14936824
 37.06293432 5.23109531]
 Variance score: 0.8511209731130607

Fig B.9 Analyzing the Linear Regression Model

```

lr_y_pred = reg.predict(X_test)
mae = float(mean_absolute_error(Y_test, lr_y_pred))
print("\nMAE: ", mae)

```

MAE: 11.27344595792351

```

[65] compare = pd.concat([pd.DataFrame(lr_y_pred), pd.DataFrame(Y_test)], axis=1)
compare.columns = ['Predictions', 'Actual Output']
compare.head()

```

	Predictions	Actual Output
0	8.388910	7.0
1	40.240787	39.0
2	4.237873	2.0
3	27.103988	52.0
4	88.626025	102.0

Fig B.10 Linear Regression Model Comparison

```

from sklearn.svm import SVR
svr_model = SVR(kernel = 'rbf')
svr_model.fit(X_train, Y_train.ravel())
y_pred = svr_model.predict(X_test)
mae = float(mean_absolute_error(Y_test, y_pred))
print("\nMAE: ", mae)

```



MAE: 18.09447204862997

Fig B.11 Mean absolute error for Support Vector Regression Model

✓
0s

```

compare = pd.concat([pd.DataFrame(y_pred), pd.DataFrame(Y_test)], axis=1)
compare.columns = ['Predictions', 'Actual Output']
compare.head()

```



	Predictions	Actual Output
0	9.523594	7.0
1	33.143646	39.0
2	13.254084	2.0
3	28.664531	52.0
4	58.912366	102.0



Fig B.12 Support Vector Regression Model Comparison

```

✓ [61] from sklearn.ensemble import RandomForestRegressor
0s rf_model = RandomForestRegressor(n_estimators = 12, random_state = 42)
rf_model.fit(X_train, Y_train)
rf_y_pred = rf_model.predict(X_test)
mae = float(mean_absolute_error(Y_test, rf_y_pred))
print("\nMAE: ", mae)

```

MAE: 7.55314371257485

```

✓ compare = pd.concat([pd.DataFrame(rf_y_pred), pd.DataFrame(Y_test)], axis=1)

```

Fig B.13 Mean absolute error for Random Forest Model

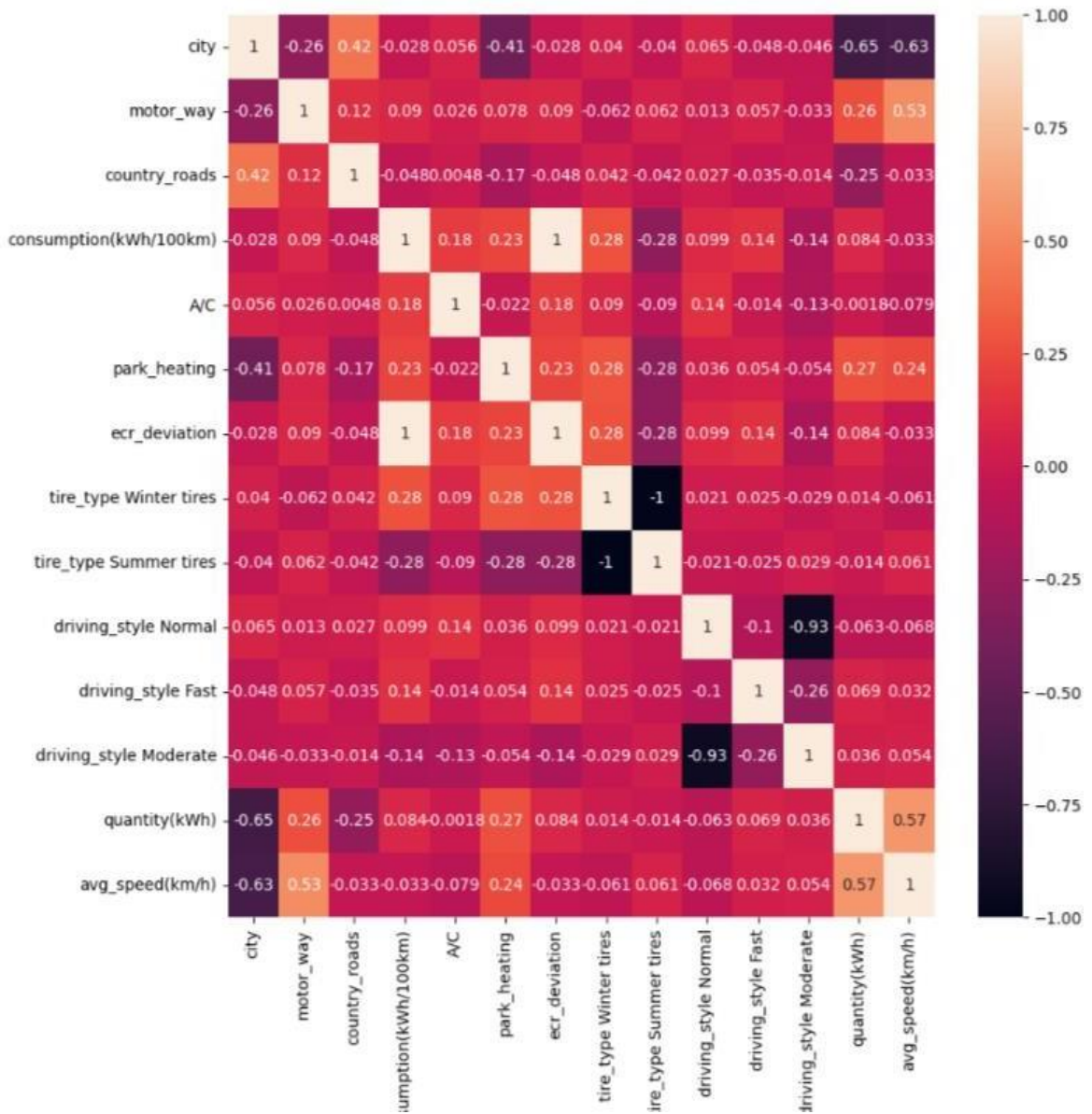
```

▶ compare = pd.concat([pd.DataFrame(rf_y_pred), pd.DataFrame(Y_test)], axis=1)
compare.columns = ['Predictions', 'Actual Output']
compare.head()

```

	Predictions	Actual Output
0	6.750000	7.0
1	41.333333	39.0
2	2.000000	2.0
3	41.416667	52.0
4	74.166667	102.0

Fig B.14 Random Forest Model Comparison



B. 15 Heatmap Comparison

C.RESEARCH PAPER

A Prognostic Approach To Alleviate Range Anxiety For E - Vehicles Using Artificial Intelligence.

Karumulla Kaushik Reddy^{1, a)}, Kodali Sai Venkat^{2, b)(*)} and Dr.S Prayla Shyr^{3, c)}

Author Affiliations

^{1,2,3}Department of Computer Science and engineering, Sathyabama Institute of Science and Technology, Chennai, Tamil Nadu, India

Author Emails

^{a)} Corresponding author: kr5645433@gmail.com

^{b)} saivenkat.kodali@gmail.com ^{c)} praylashyry.cse@sathyabama.ac.in

Abstract. Rapid growth in the transportation sector is contributing to pollution. The conventional vehicles which are made of fossil fuels emit harmful gasses that result in global warming. In this scenario e-Vehicles can be chosen as a good alternative but range anxiety is something that is stopping people from resorting to e-Vehicles. Range prediction is a crucial aspect of electric vehicles (EVs) as it directly impacts their usability and practicality. With the increasing adoption of EVs, accurately estimating the range has become essential for optimizing driving plans and addressing range anxiety. Range prediction refers to the estimation of how far an electric vehicle can travel on a single charge based on various factors such as battery capacity, driving conditions, weather, and driving style. It involves the utilization of data-driven models and algorithms that consider these variables to provide an estimate of the remaining range. The development of accurate range prediction models requires a deep understanding of EV technology, battery behavior, and real-world driving conditions. By providing drivers with reliable range predictions, they can make informed decisions, plan their journeys more effectively, and enhance their overall experience with electric vehicles. Here we propose a machine learning approach that combines physical and environmental factors to accurately predict range and therefore minimize range anxiety.

INTRODUCTION

Transport is a fundamental requirement of modern life, conventional combustion engines that use petrol and diesel are highly polluting and contribute to global warming. These conventional vehicles can be replaced with electric vehicles that have zero tailpipe emissions and are much better for the environment. They have relatively lower running costs, maintenance cost, convenience of charging at home, and are easy to drive and quiet. There is an increase in usage of electrical vehicles lately. It is expected that 145 million electric vehicles will be on the road by 2030. Range prediction is a critical factor for EV users to plan their trips effectively and alleviate range anxiety. Calculating the range of electric vehicles (EVs) using Machine Learning (ML) techniques has gained significant attention in recent years. ML algorithms leverage historical data from EVs, such as driving patterns, battery characteristics, and external factors like weather and road conditions. By analyzing this data, ML models can learn complex relationships and patterns that impact the vehicle's range. These models can then predict the remaining range based on real-time data inputs. ML-based range prediction systems can continuously adapt and improve their accuracy by learning from a large amount of data collected from various EVs and diverse driving scenarios. These models can account for factors like battery degradation, traffic conditions, driving behavior, and energy consumption patterns, providing more reliable and personalized range estimations. The integration of ML in range prediction for EVs has the potential to enhance the driving experience, optimize route planning, and increase confidence in electric vehicle technology. As research and development in this area advance, ML algorithms can contribute to maximizing the efficiency and usability of EVs, promoting their widespread adoption and contributing to a sustainable transportation future.

RELATED WORK

State of Charge (SOC)

The amount of energy that is accessible in a battery at a given moment is measured and represented as a percentage known as the state of charge, or SOC.

$$\text{SOC (\%)} = Q + \frac{Q_{\text{initial}}}{Q_{\text{max}}} * 100(1)$$

Q_{initial} = Battery initial charge.

Q = how much potency is given to or delivered by the battery. It is negative during the discharge and positive during the charge in accordance with current convention.

Q_{max} = the most amount of charge that a battery is capable of holding.

The approach that combines AdaBoost's advantages is suggested in the study [4]. RNNs are a kind of neural network that can simulate temporal relationships in time-series data, and R_t is a boosting method that can increase the accuracy of weak learners. This article [4] used a recurrent neural network to estimate the state of charge (SOC) of lithium-ion power batteries. Six input neurons, one hidden layer, and one output neuron make up the RNN utilized in this investigation. This study utilizes an RNN configuration featuring six input neurons, one hidden layer, and one output neuron.

State of Energy (SOE)

In electric vehicles (EVs), the remaining battery power is indicated by the "state of energy". State of Charge (SOC) is a metric used to measure it. The driver and the car's systems both greatly value this information. It assists in determining how much further you can drive, when to charge, and how to make efficient use of the energy. Therefore, being aware of the energy situation enables drivers to make informed decisions and maximize the usage of their electric vehicles (EVs), improving both the driving and energy-saving experience. Conventional methods of calculating a battery's state of charge (SoC) are subject to inaccuracy as they depend on an initial estimate that may not be accurate. However, there is a novel approach that combines state of energy (SOE) with SoC estimate in a creative way, as described in a framework [13]. Utilizing a unique type of software known as a multi-layer feedforward neural network, it improves accuracy even in the face of changing temperatures and driving circumstances. Test results for this approach under various driving conditions are rather encouraging, particularly when considering Mean Square Error (MSE). Put simply, it's a more dependable method of determining how much energy remains in your battery, even when things happen.

State of Health (SOH)

A battery's State of Health (SOH) indicates how well it retains charge in comparison to its original state. This percentage indicates if your battery has aged over time or is still in good condition. A higher proportion is preferable!

$$\text{SOH (\%)} = (Q(\text{max})/C(\text{rated})) * 100 \quad (2)$$

$Q(\text{max})$ = The maximum charge available of the battery

$C(\text{rated})$ = The rated capacity

In a report [3], The prolonged longevity of a battery is forecasted by considering various factors, including its ability to withstand changes in temperature and electrical flow., the number of times it has been charged, vibrations, and environmental conditions. The study discovered that machine learning (ML) approaches may yield extremely precise results, with accuracy rates as low as 5%. Some of these programs are mentioned, including Support Vector Regression, Boosted Regression Trees, AdaBoost, XGBoost, and Gaussian Process Regression. Therefore, it is similar to utilizing sophisticated instruments to estimate battery life with high accuracy.

The research [5] proposes a novel type of model (i.e., smart tool) called LSTM that incorporates attention to highlight salient features while analyzing battery data over time. They used data from thirty Li-ion batteries to test this. Their methodology integrates historical and contemporary data and employs a clever technique to identify the most relevant information. It's similar to having an improved understanding of battery behavior based on critical data from both the past and the present.

RANGE ANXIETY

Range When a driver of an electric car has anxiety, it is because they fear that the battery may discharge before they arrive to their destination or a charging station. It's similar to worrying that your electric car won't have enough battery life to get you where you need to go.

ARCHITECTURE DIAGRAM

In fig 1, the data collected through various sources is preprocessed and divided into testing and training data. An ML model is designed to calculate SOC& SOE. Again, another ML model is built which takes these calculated SOC & SOH values and other additional parameters for training and predicts the remaining range.

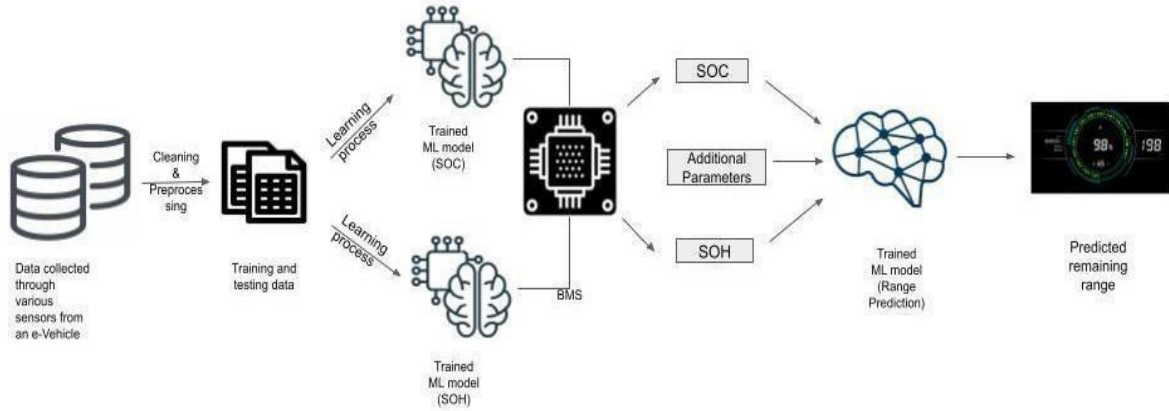


FIG 1 ARCHITECTURE DIAGRAM

According to the study [6], owners of electric cars (EVs) in India frequently experience "range anxiety." This indicates that they are worried that the battery life of their EV may not be sufficient to get them where they need to go. The concern expressed in this poll [13] is a major deterrent for more people to not convert to electric vehicles. The study's authors suggest that stronger regulations, technological advancements, and more public awareness of electric vehicles may all help to resolve issue. Therefore, they expect that by taking these steps, more consumers will have confidence while using EVs.

Influntianal Auxiliary Parameters

According to the study [8], we often use variables like voltage, current, and temperature to determine an electric car's remaining range. However, the study also makes the point that these factors alone are insufficient for reliable results. The authors emphasize that there are several other factors that might impact a car's range, including as the vehicle's design, maintenance practices, driving conditions, and battery life. According to a different research [12], factors such as temperature, battery type, and charging technique may all affect how far an electric automobile can travel. Therefore, it goes beyond the fundamentals; there are several.

Remaining Range Estimation

How far an electric car can travel before needing to be recharged is known as its remaining range. The technique described in [11] builds a model that is constantly updated with information from the vehicle's systems to anticipate how much further it can travel. As per a different research[10], the prediction of range is contingent upon the kind of computer program employed, the data it has been trained on, and the current conditions. A computer algorithm in [1] calculates the range based on driving circumstances, whereas [9] also considers temperature and traffic. In order to improve forecasts, [2] proposes combining a variety of methods, while [7] mixes computer-based and physical models while taking into account variables like battery level, road slope, and speed. Therefore, to more accurately estimate how far an electric

PROPOSED WORK

Our goal is to develop an intelligent computer software that can forecast an electric car's remaining range and battery condition. We first determine the critical variables, such as battery information, temperature, vehicle speed, and historical charging data. Next, we train the computer software (machine learning model) to produce accurate predictions using this data. We put it through tests to make sure it functions properly in various scenarios. Next, we create an additional model that incorporates more information and the battery and energy estimations from the first model to forecast the range of the electric vehicle between charges. It's similar to teaching a computer how to recognize a battery and determine how long you can drive.

State of Charge (SOC)

1. Data Accumulation: Utilizing drive cycle data obtained from a Panasonic 18650PF Li-ion battery at 25°C, a power profile was calculated for the electric Ford F150 truck's driving cycle. Drive cycle data obtained from the Panasonic 18650PF Li-ion battery of an electric vehicle (EV) operating under ambient conditions of 25°C was utilized. Subsequently, a calculation was performed to determine the driving cycle power profile for an electric Ford F150 truck.

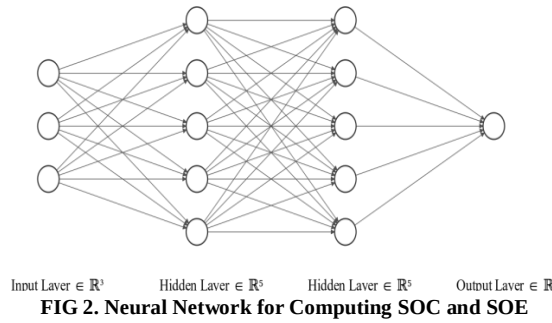
2. Pre-processing: We extracted and cleaned the dataset during the data preparation stage, adding characteristics like Ah, Wh, Current, Voltage, Battery temp, Chamber temp, time, and power. To ensure consistency in the training of the model, we eliminated outliers and normalized the data. In particular, we retained the three most important variables voltage, temperature, and current—while eliminating all other aspects in order to calculate the State of Charge (SOC). By multiplying pertinent variables by a notional amp-hour (2.5Ah) and dividing the result by 2.5Ah, the SOC was computed. This methodical procedure guarantees the accurate creation of SOC, which illustrates the gradual drain of the battery.

3. Data Analysis: Since there are no missing values in the dataset and no category values, encoding is not required, and imputation is not necessary. There were no discernible positive or negative relationships between the characteristics according to the Pearson correlation coefficient. Using the Z-Score test, outliers were found and eliminated. After completing these preparation stages, the dataset is now error-free and prepared for training, laying the groundwork for additional analytical processes.

4. Data Splitting: 20% of the data is utilized for testing, while the remaining 80% is used for training.

State of Energy (SOE)

The same procedures as previously are used for data collection and preparation, but we see SOC as an extra feature



The neural network (Fig. 2) has two hidden layers with five neurons each, an output layer with a single neuron, and three neurons in the input layer. Relu, defined as $\max(0, x)$, is employed as the activation function for the hidden layers, with x serving as the input. In other words, a positive input stays intact, while a negative input becomes zero. During training, this feature helps the network discover and recognize complex patterns in the data.

$$f(x) = \max(0, x) \quad (3)$$

and a linear activation function, which has a mathematical definition in (4), is employed for the output layer.

$$f(x) = x \quad (4)$$

Early halting and the use of an Adam optimiser are employed to stop the model from overfitting.

Remaining Range Estimation

1. Data Accumulation: Real driving data from two Volkswagen e-Golf vehicles built between 2014 and 2016 was used to train our model. The collection contains information from 3615 journeys that totaled around 152,167 kilometers. The model may learn and make predictions based on the real experiences of these particular electric cars while they are traveling thanks to this practical data.

2. Pre-processing: Key characteristics were standardized and encoded to get the data ready for training. A/C, park heating, average speed, fuel type, tire type, city, highway, rural roads, driving style, consumption, manufacturer, model, version, power, fuel date, and ECR derivation were among the details included in the dataset. Certain information was left out, including the gasoline type, date of fuel, power, manufacturer, model, and version. The remaining data was transformed into the proper formats so that the model could comprehend it. The data is now arranged such that the model can learn from it effectively.

3. Data Analysis: We eliminated records that had missing values for the primary prediction. If any information was lacking for additional details, we substituted average or middle values. The data was then partitioned into two categories: continuously variable variables (such as distance, quantity, and speed) and randomly variable variables (such as city, highway, and air conditioning). We used histograms to examine the distribution of smooth objects. Graphs were utilized to examine the relationship between the jumpy variables and our desired prediction. This aids in our improved comprehension of the facts prior to forecasting.

4. Encoding: Using one-hot encoding, categorical features like driving_style and tyre_type are encoded.

5. Outlier analysis: We examined the continuous characteristics in the hopes that their distribution would resemble a regular one. The data points that were significantly out from the average—that is, more than three times or less than three times the typical spread—were then eliminated. Eliminating these extreme points, often known as outliers, makes our data more suitable for study.

6. Correlation b/w features: We used a heat map to examine the relationships between various characteristics. It demonstrated the close relationship between the "Normal" and "Moderate" driving styles, however having both is a bit excessive. Thus, we opted to retain only one and eliminated "Normal." Similarly, while "Summer" and "Winter" tire types go hand in hand, we only need one, thus we eliminated the "Summer" tire type. This streamlines matters without omitting crucial details.

7. Data Splitting: 20% of the data is utilized for testing, while the remaining 80% is used for training.

In the Fig. 3. there are twelve nodes in input layers

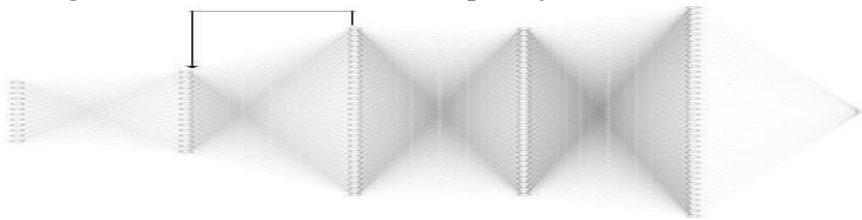


Fig 3: Neural Network for Estimating Remaining Range

The dense layers consist of two layers, with the first layer comprising 32 nodes and the second layer comprising 64 nodes. The Tanh activation function, as outlined in reference (5), is employed for the LSTM layers.

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Following (3) meticulously, ReLU is meticulously implemented within dense layers to achieve the desired output, consolidated through a solitary node forming the output layer, all underpinned by the use of a linear activation function.

ANALYSIS

The analysis of the dataset establishes the association between different parameters.

A) Distance covered under different conditions.

1) Distance comparison in city roads

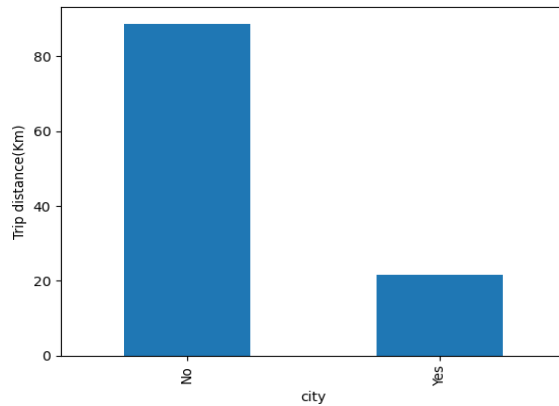


Fig 4: Travel time compared to city roads

In Figure 4. In cities, the journey distance is shorter (This may be because of traffic circumstances that waste energy by forcing the vehicle to use the brakes rather than regenerative braking.)

2) Travel duration compared to highways

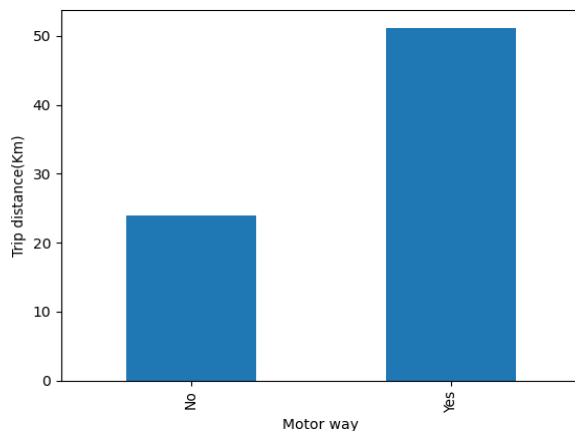


Fig 5: Comparison of travel time on roads

The trip distance on motor routes is greater in Fig. 5. (This could be as a result of motorways being free, which results in steady speeds, fewer stops, and less accelerations.)

2) Travel time in comparison to rural roads

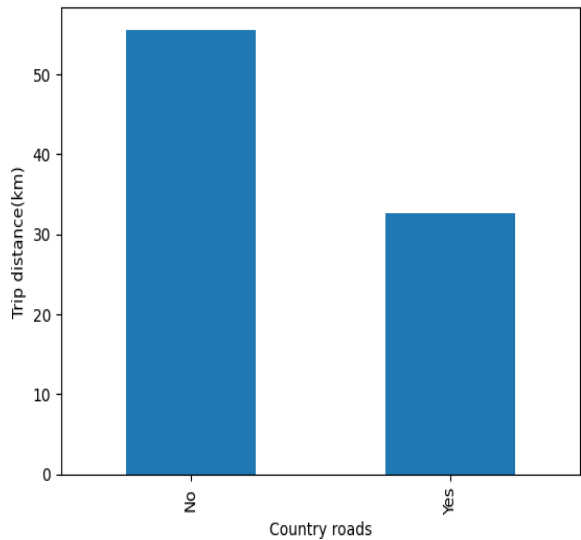


Fig 6: Comparison of travel time on roads

In Figure 6. Travel distance is shorter on country roads (perhaps due to their uneven surface, bumps, and diversions, which result in variable speed restrictions, frequent accelerating and stopping, and sparse infrastructure for charging).

4) Information on car batteries in relation to travel distance

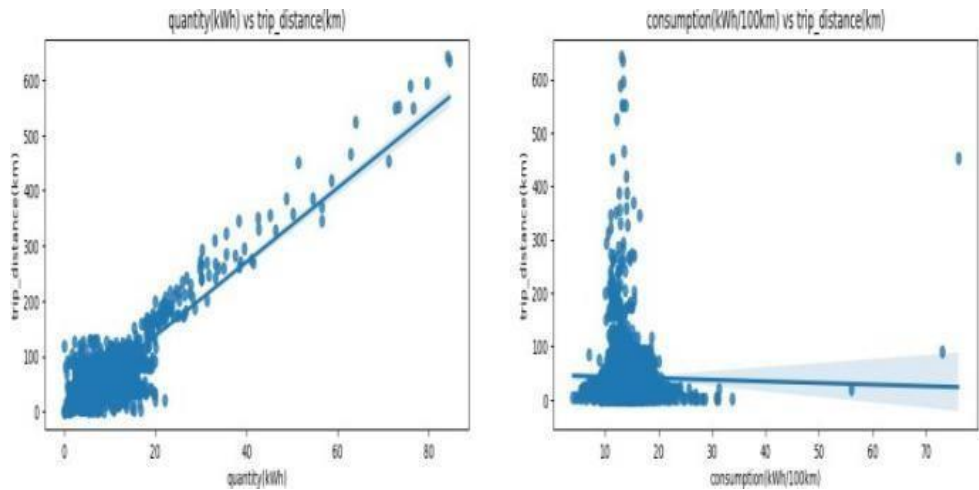


FIG:3.4 Plot of journey distance vs quantity and consumption (regression)

3)Plot of average speed against trip distance regression

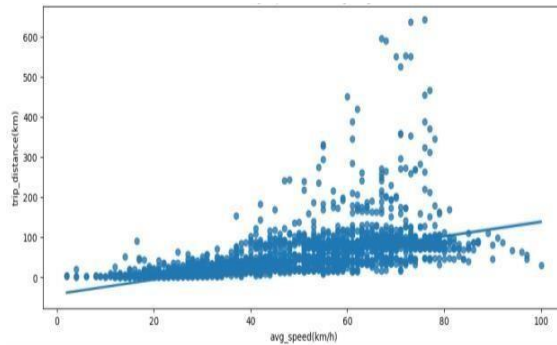


Fig 7: Regression plot of average speed against trip distance

in Figure 3.5. Since the data points show an upward trend up to 80 km/h, the average speed will be taken into account when determining the distance range.

A. How supplemental devices affect consumption

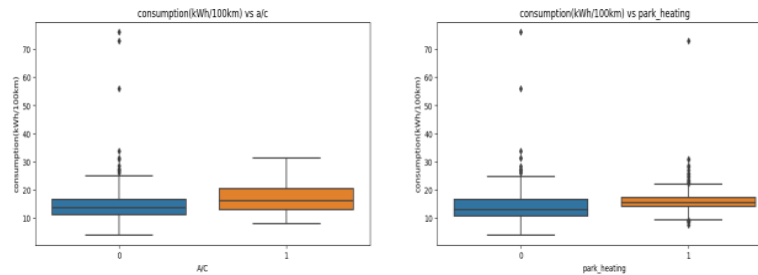


Fig 8: Comparison of vehicle consumption with auxiliary devices

Figure 8 illustrates how using the A/C increases energy use, and how using park heat increases battery usage, demonstrating the impact of different climate control systems on overall energy consumption in the vehicle

RESULTS

This chapter discusses the outcomes of the suggested remedy.

A. SOC

1)Comparison between expected and actual values

	Predictions	Actual Output
0	0.135118	0.167040
1	0.793014	0.791524
2	0.249730	0.254800
3	0.279199	0.282960
4	0.816404	0.790392
...	...	...
20484	0.147491	0.162160
20485	0.984723	0.928672
20486	0.316771	0.315720
20487	0.878058	0.884896
20488	0.284465	0.284760

20489 rows × 2 columns

SOC expected vs. actual values in Figure 9

Comparing expected and actual values shows where predictions differ from reality, helping refine forecasts and make better decisions. This analysis improves planning and risk management by spotting differences and enabling adjustments for future predictions.

2) Error

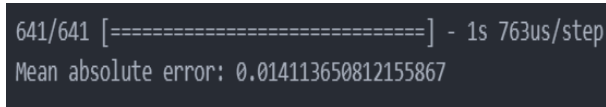


Figure 10 shows the SOC model's mean absolute error.

Figure 10 illustrates the mean absolute error of the SOC model, indicating the accuracy of battery charge predictions, crucial for optimizing electric vehicle efficiency.

B.SOE

1)Comparison between expected and actual values.

	Predictions	Actual Output
0	0.161298	0.161857
1	0.777971	0.777673
2	0.244460	0.244004
3	0.271357	0.271443
4	0.777336	0.776499
...	...	...
20484	0.157434	0.157908
20485	0.920528	0.921808
20486	0.302690	0.302908
20487	0.874473	0.874161
20488	0.272930	0.273210

20489 rows × 2 columns

SOE expected vs. actual values in Figure 11

Figure 11 compares expected and actual State of Energy (SOE) values, offering insight into the accuracy of energy level predictions, vital for evaluating electric vehicle energy management systems

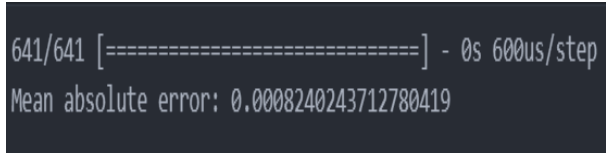


Figure 13 shows the SOE model's mean absolute error.

Figure 13 displays the mean absolute error (MAE) of the SOE model, providing insights into energy level prediction accuracy, crucial for evaluating electric vehicle systems.

C. Remaining range estimation

1) Comparison between expected and actual values

	Predictions	Actual Output
0	19.913588	20.0
1	19.177187	19.0
2	20.933769	22.0
3	20.233088	20.0
4	19.742170	20.0
...	...	...
662	142.805740	143.0
663	23.974070	12.0
664	4.279182	5.0
665	20.014788	20.0
666	31.427786	33.0

667 rows × 2 columns

Figure 14: Projected vs. actual values for the remaining range

Contrasting projected with actual remaining range in electric vehicles aids in evaluating estimation accuracy. This informs drivers for efficient travel planning and behavior adjustments. Understanding differences between projections and actual values refines range prediction algorithms, contributing to improved vehicle performance and usability

1)Error

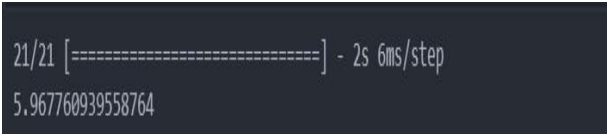


Fig 15 displays the mean absolute error for the remaining range estimate model.

In Fig7.2 showcases the mean absolute error pertaining to the remaining range estimation model. It provides a visual representation of the model's accuracy in predicting the remaining range.

3) Gradual loss of validation

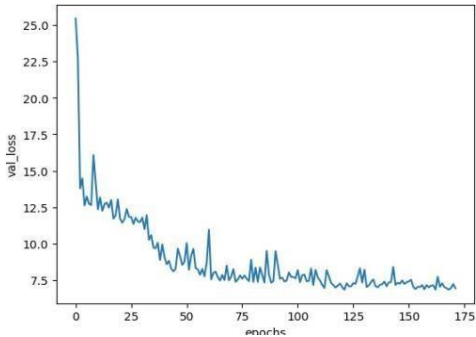


Fig: 16 Validation loss as a function of epochs

The validation loss depicted in Figure 7.3 commenced around 25.0 and steadily declined to 7 over the course of 175 epochs, indicating improved model performance with iterative training.

3) The average absolute error throughout time.

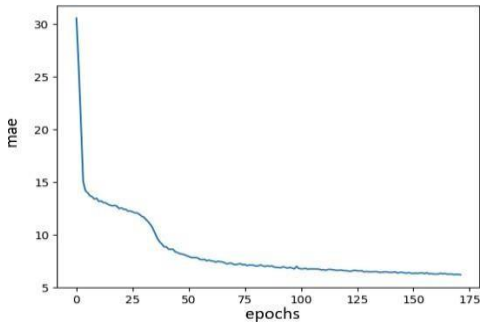


Fig 17: Mean absolute error across epochs

After 175 epochs, the mean absolute error (MAE) depicted in Figure 7.4 dramatically decreases from an initial value of approximately 30.0 to a significantly lower value of 5, showcasing the model's progressive improvement over iterations.

D. Evaluation of alternative models.

Box I: Exam outcomes compared to additional Paradigms.

Paradigms	Mean deviation	Criteria
Support Vector Regression	15.25	Core = rbf
Line of bestfit analysis	10.28	Polyvalent
Stochastic woodland	6.78	Estimaation count = 12
Multilayer Perceptron	5.96	Group size = 16 Enhancer =Root Mean Square Update Signal handlers = Timely halting

It is evident by above that MLP has the lowest mean absolute error. Thus, MLP is the most trustworthy method for estimating remaining range.

The average difference between the computed and real values is determined using Mean Absolute Error. It is defined formulae in (5)

$$MAE = \sum \frac{|y_i - x_i|}{n} \quad (6)$$

In (6), where y_i was observation's actual value and x_i was n is the total number of observations, and the expected value for i th observation .

FUTURE WORK

The new model is all about guessing how much more an electric car can travel. It looks at things like how steep the road is, how the driver behaves, the weather, and other gadgets in the car. The model also knows that having more people in the car can change how far it can go, and in the future, it can learn even better if we add this as a rule. This way, the model gets better over time at telling you exactly how much distance your electric car can cover.

CONCLUSION

Excellent results were obtained using the machine learning model that we employed to determine the battery's State of Charge, or SOC. The model produced precise predictions when we utilized these SOC values to estimate the remaining energy (also known as the State of Energy, or SOE). 5.96 is a very low Mean Absolute Error (MAE), which indicates how well our estimates match the real numbers. This indicates that our approach is well suited for practical use, offering trustworthy approximations of the battery's remaining capacity and, hence, the range of the electric car.

References

1. Penghua Li, Zijian Zhang, Qingyu Xiong, Baocang Ding, Jie Hou, Dechao Luo, Yujun Rong, Shuaiyong Li, (2020) "State-of-health estimation and remaining useful life prediction for the lithium-ion battery based on a variant long short term memory neural network", *Journal of Power Sources*, Volume 459
2. B. Zheng, P. He, L. Zhao and H. Li, (2016) "A Hybrid Machine Learning Model for Range Estimation of Electric Vehicles," 2016 IEEE Global Communications Conference (GLOBECOM), Washington, DC, USA, pp.1-6
3. P. Sharma and B. J. Bora, (2022) "A Review of Modern Machine Learning Techniques in the Prediction of Remaining Useful Life of Lithium-Ion Batteries," *Batteries*, vol. 9, no. 1, p. 13
5. R. Li, H. Sun, X. Wei, W. Ta, and H. Wang, "Lithium Battery State-of- Charge Estimation Based on AdaBoost.Rt-RNN," *Energies*, vol. 15, no. 16, p. 6056, Aug. 2022
6. Ganesh Sankaran, S. Venkatesan, M. Prabhakar, Range Anxiety on electric vehicles in India -Impact on customer and factors influencing range Anxiety, *Materials Today: Proceedings*, Volume 33, Part 1, 2020
7. Miri, I, Fotouhi, A, Ewin, N. Electric vehicle energy consumption modelling and estimation—A case study. *Int J Energy Res.* 2021
8. Szumska, E.M.; Jurecki, R.S. Parameters Influencing on Electric Vehicle Range. *Energies* 2021, 14, 4821.
9. Jun Bi, Yongxing Wang, Qiuyue Sai, Cong Ding, estimating remaining driving range of battery electric vehicles based on real-world data: A case study of Beijing, China, *Energy*, Volume 169, 2019, Pages 833- 843, ISSN 0360-5442
10. JOUR, Sun, Shuai, Zhang, Jun, Bi, Jun, Wang, Yongxing, 2019/01/09, A Machine Learning Method for Predicting Driving Range of Battery Electric Vehicles, 4109148
11. J. Hong, S. Park and N. Chang, "Accurate remaining range estimation for Electric vehicles," 2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC), Macao, China, 2016
12. M. Smuts, B. Scholtz and J. Wesson, "A critical review of factors influencing the remaining driving range of electric vehicles," 2017 1st International Conference on Next Generation Computing Applications (Next Comp) Mauritius, 2017
13. S. Sharma, P. D. Achlerkar, P. Shrivastava, A. Garg and B. K. Panigrahi, "Combined SoC and SoE Estimation of Lithium-ion Battery using Multi-layer Feedforward Neural Network," 2022 IEEE International Conference on Power Electronics, Drives and Energy Systems (PEDES), Jaipur, India, 2022, pp. 1-6
14. O. Rahbari, N. Omar, P. Van Den Bossche and J. Van Mierlo, "A centralized state of charge estimation technique for electric vehicles equipped with lithium-ion batteries in smart grid environment," 2018 IEEE International Conference on Industrial Technology (ICIT), Lyon, France, 2018, pp. 1721-1725
15. P. A. Topan, M. N. Ramadan, G. Fathoni, A. I. Cahyadi and O. Wahyunggoro, "State of Charge (SOC) and State of Health (SOH) estimation on lithium polymer battery via Kalman filter," 2016 2nd International Conference on Science and Technology-Computer (ICST), Yogyakarta, Indonesia, 2016, pp. 93-96
16. M. Xu, Q. Zhu and M. Zheng, "A State of Charge Estimation Approach Based on Fractional Order Adaptive Extended Kalman Filter for Lithium-ion Batteries," 2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS), Enshi, China, 2018, pp. 271-276

D.CERTIFICATE



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
CATEGORY-1 UNIVERSITY BY UGC
Accredited 'A++' Grade by NAAC | 2B Status by UGC | Approved by AICTE



www.sathyabama.ac.in

SCHOOL OF COMPUTING
Department of Computer Science and Engineering

Certificate of Presentation

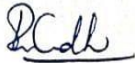
This is to certify that
KARUMULLA KAUSHIK REDDY
of
Sathyabama Institute of Science & Technology
has presented a paper entitled
A Prognostic Approach to Alleviate Range Anxiety for E-Vehicle Using AI
in the
International Conference on Cognitive Computing and Artificial Intelligence (ICCCAI-2024) in association with
Taylor's University, Malaysia held on 7th & 8th March 2024.



Dr. Sayan Kumar Ray
Head, School of Computer Science
Taylor's University



Dr. L. Lakshmanan
HOD, CSE Dept
Sathyabama Institute of Science and Technology



Dr. T. Sasikala
Dean, SOC
Sathyabama Institute of Science and Technology