SMART ELECTRIC VEHICLE CHARGING SYSTEM BASED ON TIME-OF-USE TARIFF, HOUSEHOLD LOAD AND USER DRIVE CYCLE DATA

H. Chen

3rd Year Project Final Report

Department of Electronic &
Electrical Engineering

UCL

Supervisor: Dr Ryan Grammenos

30 March 2022

I have read and understood UCL's and the Department's statements and guidelines concerning plagiarism.

I declare that all material described in this report is my own work except where explicitly and individually indicated in the text. This includes ideas described in the text, figures and computer programs.

This report contains 36 pages (excluding this page and the appendices) and 11988 words.

Signed: _____Henry Chen_____ Date: _____30.03.2022_____

(Student)

Acknowledgement

This report has been developed in collaboration with IntelliCasa Ltd, UK.

Special thanks go to Elie for helping to get remote access to a Tesla vehicle.

And Kostas for aiding the process of obtaining data using the CAN logger from a BMW i3.

Contents

SMART ELECTRIC VEHICLE CHARGING SYSTEM BASED ON TIME-OF-USE TARIFF, HOUSEHOLD LOAD AND USER DRIVE CYCLE DATA

H. Chen

Abstract

The rate of adoption of electric vehicles (EVs) in 2020 has increased by 186% since 2019. This rapid increase can cause significant strain on the grid which in order to support this, would require huge amounts of investment to upgrade current grid infrastructure. Furthermore, many are reluctant to purchase EVs due to range anxiety.

This project proposes an automated smart EV charging system (Recommendation Module) that suggests optimal charging periods to suit the user while minimising impact on the grid. To achieve this, a collection of household load, time of use tariff and user drive cycle data are obtained which is then forecasted 7 days into the future to simulate future energy consumption and prices. The Recommendation Module then finds the cheapest, most convenient periods to charge the user's EV. Results show that this charging system is able to reduce charging costs by 47% over a period of 7 days. This demonstrates a highly efficient system which can significantly reduce the cost of charging for users, while also reducing the impact of charging on the grid.

All the code developed for this project is available at https://github.com/Henchen99/BEng-Project.git

Nomenclature – Acronyms:

EV	Electric Vehicle
SOC	State of Charge
TOU	Time of Use
API	Application Programming Interface
VED	Vehicle Energy Dataset
OBD	On Board Diagnostics
DBC	DataBase Container
ARIMA	Auto-Regressive Integrated Moving Average
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
MAPE	Mean Absolute Percentage Error
MPE	Mean Percentage Error

Nomenclature – Key Terms:

Smart Charging	Controlled Charging
Dumb Charging	Uncontrolled Charging
Aggregator	An intermediary between EV customer and energy distributors
Centralised system	Primary control and decision making of the whole system occurs at a central node
Decentralised system	Primary control and decision making of the whole system occurs at individual lower level nodes
Recommendation Module	System developed in this project to recommend optimal EV charging periods for the user

1 Introduction and Literature Review

1.1 Introduction

The UK government plans to fully decarbonise transport and achieve net-zero by 2050, with one of the main focuses being zero emissions for cars and vans by 2035 [1][2]. Current figures show that the domestic emissions of cars in the UK make up 68% of total greenhouse gas emissions by transport [3][4], making it the biggest contributor in the sector. The transition to electric vehicles (EVs) is, therefore, a significant step to achieving net-zero emissions. By 2030, the UK plans to end the sales of all petrol and diesel cars, and ideally by 2050, EV cars will represent 100% of all the cars driving on the roads [5]. In 2020, 108,000 EV cars were sold in the UK, which is a 186% increase from the year before [6][7].

Unfortunately, such rapid penetration can cause significant disruptions on the infrastructure because of stress put on the power grid due to 'dumb', or uncontrolled, charging methods [8][9], which is when users typically charge their vehicles immediately when they get home. However, this is often during peak demand which can overload the grid and may result in power cuts and blackouts. Study [10] has shown that a 60% market penetration of EVs would require a 15% increase in investment costs to infrastructure and increase energy losses by 40%. If users continually use this method of charging, huge investments into the upgrade of electricity infrastructure would be required to support this change.

Therefore, a current solution to this problem, recommended by the UK government is 'smart charging' to reduce charging at peak times [11][12]. Energy suppliers now offer installation of smart meters that monitor electricity usage and provide dynamic time-of-use (TOU) tariffs so that the price of electricity fluctuates during the day, with the highest price during peak times, and the lowest typically at night [13]. With this system, customers are rewarded with cheaper charging costs when charging their vehicle at off-peak times, thus inadvertently reducing the peak loads put on the grid.

This solution however is quite unreliable as it requires significant user input. Users would need to be both cost and time-conscious to actively remember to plug in the vehicle later at night rather than the more convenient practice of charging immediately once they are home. If a vehicle owner is unwilling or unable to act in this way, the solution would not work. In addition, this method does not solve the other key issues with EV cars, such as range anxiety, sustaining battery health, and general ease of use of the vehicle. Therefore, the implementation of an automated smart charging system [11] will be a key factor in the successful rollout of EVs.

This project aims to develop a smart charging algorithm that can address several of the issues listed above. The system looks to reduce peak charging demand by up to 80% [14], reduce charging costs by up to 40% [15], and maintain and improve battery longevity, ideally keeping battery state of charge (SOC) in the 20-80% range [16]. By also taking into account user drive cycle data and time-of-use tariffs, the algorithm will calculate ideal charging times for the user so that they do not need to worry if the current SOC will be sufficient for the next journey, providing a convenient, seamless transition into ownership of an EV.

This report is structured as follows: Section 1.2 is a comprehensive literature review of current smart EV charging systems. It will also set the goals for this project and define how they will be achieved. Section 2 will then provide the fundamental background theory necessary to understand the work done in this project. Section 3 will explain the methodology and work done within this project to implement a smart EV charging system. This is followed by the main body of the report, Section 4, which discusses and critically evaluates the project's results and findings. Section 5 will conclude the full report as well as provide ideas for future work that can expand and improve on the findings established in this project. Additional information regarding the project can also be found in Section 7, the Appendix.

1.2 Literature Review

There has been an abundance of research done in the field of EV charging scheduling systems. Simulations for different EV charging strategies have been performed to analyse and understand the impact they can have on the electricity grid [17]–[21]. From this research, there is conclusive evidence that controlled charging can outperform uncontrolled charging methods in reducing electricity charging costs by up to 70% [20][21] while also minimising the increase in peak load. Furthermore, most of these systems can be divided into two main categories: centralised or decentralised [22][23].

With centralised systems, a central controller monitors all charging points to calculate and determine the charge rate that should be delivered to each individual EV on the system [24][25] (Figure 1). This gives centralised systems its main advantage: [26][27], they are able to regulate and ensure charging rates meet customer demands, allowing aggregators to gain maximum profits, all while working within network distributor constraints. Some papers have used an Adaptive Charging Network to support large scale EV charging, prioritising operator costs and have been profitable up to 3.4 times compared to uncontrolled charging [28]. However, centralised systems have poor scalability, and its implementation is computationally complex due to the need for constant communication between an aggregator and EV, and access to large amounts of global user data [22][27][29]. Furthermore, its priority in fairness-based charging, over maximising the gains of individual consumers [27] and the decrease in user privacy [29] currently makes it less popular system for EV owners, meaning it would be difficult to implement as a short-term solution.

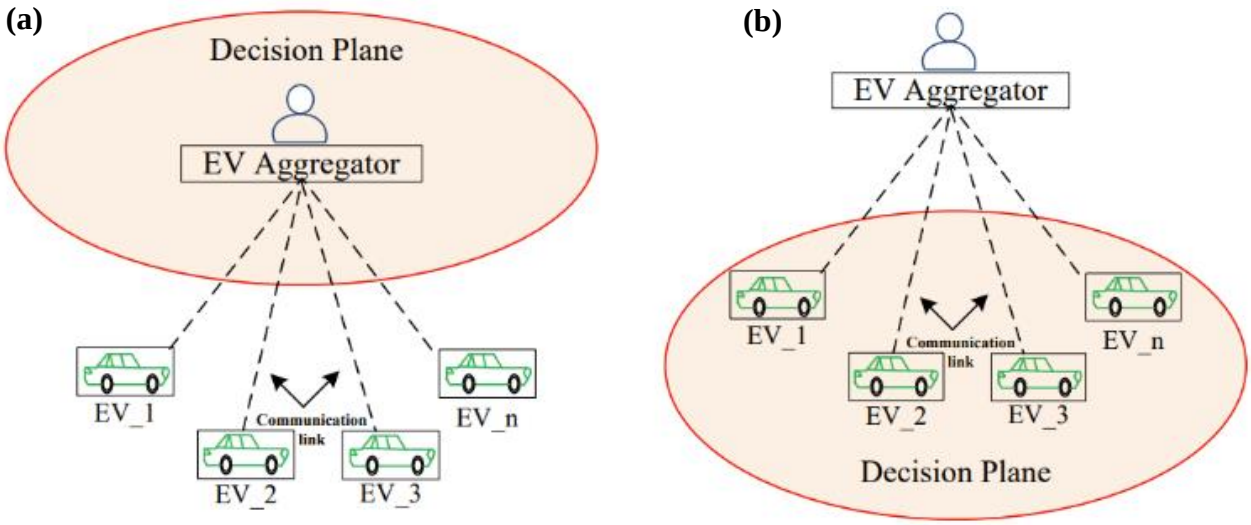

Figure 1 – Visual comparison of Centralised (a) vs Decentralised EV charging control (b) [25]

On the other hand, decentralised systems, also known as distributed control, allow individual systems (EVs or charging stations) to make their own decisions based off common data [30][31]. Algorithms are also implemented offline, which significantly reduces computational complexity compared to centralised systems. Typically, decentralised algorithms have a narrower aim on a particular area they want to optimise, for example, they can be grid, aggregate, EV, or consumer oriented [19][23][32]. This can be suboptimal overall, since

aggregators cannot directly regulate charging conditions [19][32], but with the current state of EV technology and charging infrastructure, as well as the need for rapid EV penetration in the next few years, decentralised charging systems are a faster, and easier-to-scale solution. [24][26] Present aggregate-orientated solutions, aims for local individual charging stations to produce the best charging rates for the vehicle while maintaining local grid conditions and minimising power supply costs. Grid-orientated solutions such as [26][33][34] involve load flattening/valley filling, where they aim to minimize power load variation throughout the day. However, research has shown that current consumer concerns of smart charging are over energy costs and range anxiety [35][36][37]which makes these charging systems less popular.

A more common solution of decentralised charging is consumer-oriented EV charging, in particular, minimising charging costs [20][23][38]. Presented solutions vary in approach where some allow the user to submit price preferences and travel schedules to minimize their charging costs [38], while others use peak-power based tariffs, which have been able to reduce charging costs by 59.8% [20]. In addition, dynamic TOU tariff pricing and grid conditions are usually directly related [39] which means that, reducing the cost of charging will also reduce the impact on the grid. What these solutions do not consider however, is convenience for the user. In the UK on average, 72% of vehicle owners have off-street parking, i.e. own a driveway or garage, but in densely-populated locations such as London, only 48% have off-street parking, which makes daily charging inconvenient [40][41]. For EV owners without off-street parking, finding the best charging periods every few days instead of daily would be more realistic. However, to be able to allocate ideal charging periods a few days in advance, user driving behaviour needs to be understood. This is possible through the use of user drive cycle data such as vehicle SOC or speed [42], collected directly from the vehicle itself. The papers mentioned previously have all used simulated vehicle data, but real-world data is unpredictable, so for the algorithm to be implemented accurately, data from real-life users need to be used.

One paper which proposes a similar smart charging system solution to this project is the work of Shyang et al [42], an autonomous, residential, distributed EV charging system. In this paper, they develop a smart EV charging system that takes into account EV telemetry data (SOC, vehicle speed) alongside other typically seen data such as TOU tariff and household load data to produce an optimal charging system. This novel approach allows the system to be more centred around the EV user themselves so that the user can reap the maximum benefits of smart charging from their EV. However, a limiting factor in their implementation is the lack of real SOC data. This means they also needed to simulate this data which, although produced valid results, may not be entirely realistic or representative of human behaviour.

Previous work has shown there is greater short-term advantage for decentralised systems, as the rate of EV adoption is rapidly increasing each year, a quick to implement, power grid and user-friendly solution is required. However, as can be seen with current centralised and decentralised implementations, there is a lack of real EV data which would allow us to accurately determine how smart charging will affect the EV charging infrastructure. This project will extend the work of Shyang et al [42] to acquire this data before integrating it into current charging algorithms.

Goals and Objectives:

This project aims to improve upon existing work by following the goals and objectives set below.

- Obtain EV User Drive Cycle Data, Household Load profile Data and TOU data
    - Collect continuous EV user drive cycle data (SOC, vehicle coordinates, speed) for a minimum of 1 week, logged through either a CAN bus data logger, pulled using an API, or datasets published online
    - Browse online for published/peer-reviewed datasets to find UK based, household load datasets for a continuous period of at least 1 month
    - Obtain dynamic TOU tariff data from UK energy companies for the same time period found from the household load data

- Data Extraction and Analysis of the obtained data
    - Clean obtained data (outliers or erroneous results removed if justified) using any preferred software/programming language, then format it so that missing data is forward filled, and time scale intervals are suitably set for each dataset (1 second, 1 min, 30 min)
    - Visualise the data through appropriate graph plots using any preferred software/programming language and determine trends/timestamps/key points such as off-peak times, cheapest energy periods, amount of power an empty house draws, driving routines and behaviours

- Develop Smart Charging Algorithm
    - Forecast energy prices for TOU tariff and household load data using time-series forecasting models for at least 2 days. Models used can include: Autoregressive Integrated Moving Average (ARIMA/SARIMA), Facebook Prophet (FBProphet). Forecasts are evaluated using forecasting metrics (MAE, MSE, MAPE), comparing them to determine which is the most optimal model for forecasting each dataset
    - Develop an algorithm following the logic of the recommendation algorithm flowchart (Figure 6) to find optimum charging periods based on all three obtained datasets to minimise charging cost, matching or improving on the result of a previously implemented smart charging system [42] which achieved 37.8% reduction in charging cost

2 Background Theory

This section will provide the knowledge needed to understand the methodology of implementing the Recommendation Module.

2.1 User Drive Cycle Data

Drive cycle data is the behaviour of a particular vehicle parameter over time. This can be vehicle speed, engine RPM, and battery SOC. For this project, it is vital that real-time user drive cycle data, particularly SOC, is collected as this will help give an understanding of an EV owner's charging routines. This project uses two different methods to extract data from an EV: a CAN bus data logger and an API.

2.1.1 CAN Bus Data Logger

Since 1996, OBD2 has been mandatory on all modern vehicles [43], so an intuitive method of obtaining EV data is using a CAN bus data logger.

Controller Area Network (CAN) bus

The CAN bus is a serial communication bus responsible for all communications within vehicles [44]. It allows the vehicle's internal electronics to communicate with each other and is a centralised location for diagnostic and updating of software [45]. All electronic control units (ECU), each control a specific function of the vehicle such as engine, airbags, sensors, and so forth, are connected via the CAN bus as individual nodes, thus reducing the complexity of the hardware and increasing robustness (Figure 2).

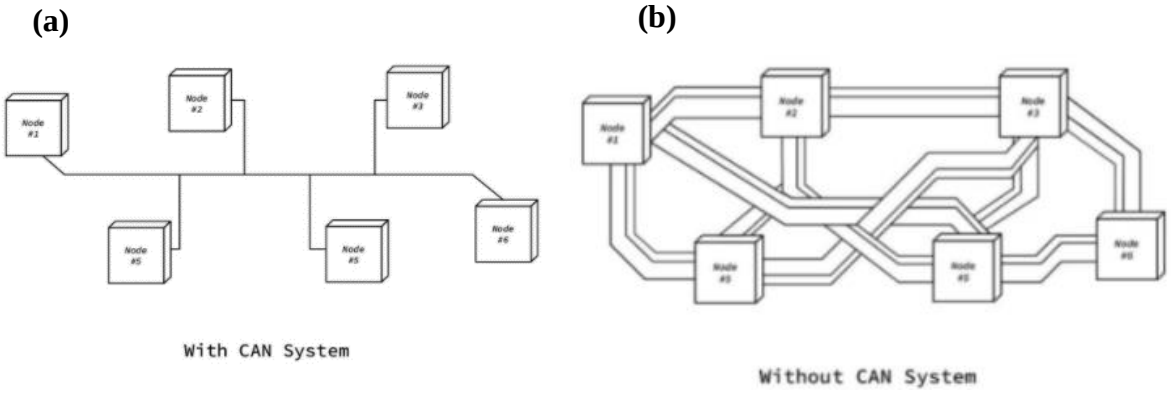

Figure 2 – Comparison of CAN (a) vs non-CAN system (b) [44]

CAN frames are what is used to communicate over the CAN bus. Raw CAN data is not easily readable and needs decoding using DBC files. Since 2008, all modern cars use the ISO15765-4 protocol [46] which is the standard for sending data-packets over the CAN bus. A diagram of this message structure can be found in Appendix 7.1.

On-board diagnostics (OBD)

All modern cars support OBD2, which is a vehicle self-diagnostic system [47]. OBD2 records driving patterns, diagnoses the 'health' of the vehicle, and detects and reports current problems. It can be accessed through an OBD2 port, which is typically found under the steering wheel in

a 6, 9 or 16 pin configuration [48](Appendix 7.2). Using OBD2 data loggers, the vehicle's data can be accessed as it logs raw CAN frames produced by the car. In addition, specific vehicle information can be requested through OBD2 using PIDs. There are a few standard PIDs [49] however, car manufacturers may also define their own PIDs, so the challenge is finding the correct PID for the data needed.

Database Container File (DBC)

DBC files are text files that allow for the decoding of raw CAN messages into a human-readable format (physical values). Every CAN ID must be matched to its respective DBC ID to ensure the correct conversion of the data [49][50] and by doing this, desired signals can be extracted. The DBC message structure can be found in Appendix 7.3.

2.1.2 Application Programming Interface (API)

As vehicle software systems become increasingly complex, car manufacturers are starting to continuously monitor vehicle data to improve safety and reliability. They can also use the data to improve the vehicle. For example, Tesla collects data from all its vehicles to improve their self-driving technology [51].

APIs are used as to achieve this as they enable large-scale data distribution in a secure and controlled manner. They function as software intermediaries to allow two applications to communicate with each other, whether it is a vehicle to manufacturer headquarters or the vehicle's in-built GPS, which would require a location API [52]. They also provide many benefits to the user because car owners can use them to operate many vehicle functions through internet-connected mobile devices [53]. Often APIs need to distribute or request for sensitive data and consequently, security is extremely important [53][54]. APIs can provide this protection using authentication tokens, which are uniquely generated codes lasting a brief period of time so that applications can access secure information.

2.2 Time Of Use (TOU) Tariffs

Energy tariffs come in several types, the two most common being fixed, or TOU [55][56].

A fixed tariff is where the cost of energy is the same for the duration of the contract, which can allow the consumer to budget their energy usage in advance. However, since there are no cheaper energy prices, it provides no incentive for the consumer to user more energy at off-peak times [57].

As a result, many energy suppliers have introduced TOU tariffs to reduce carbon emissions produced by fuel and coal-burning power plants. As the energy demand naturally fluctuates throughout the day (Figure 3) with peak periods usually from 4-7 pm [13][57][58], TOU tariffs reflect this with changes in prices for the consumer. This provides an incentive for consumers to use more energy at off-peak times, paying less for energy and therefore reducing peak demand, thus 'flattening the peak' [59]. TOU tariff can be split into Economy 7/10 or dynamic.

Economy 7/10 is a variable rate tariff that provides a cheaper fixed energy price at off-peak times for 7 or 10 hours respectively, but more expensive fixed energy cost during the day. This

can benefit EV owners as the cheap energy costs are usually set at night-time when the user is not driving [60].

Dynamic TOU tariffs frequently update the different price of energy throughout the day, updating for example, hourly or every 30 minutes. For dynamic tariffs, the cost is highly dependent on the current state of the grid, so off-peak times can be cheaper than fixed or economy 7/10 tariffs, but peak times can be much more expensive [61]. This tariff is ideal for EV owners as they can take advantage of cheapest energy costs available and is the tariff strategy this project will use.

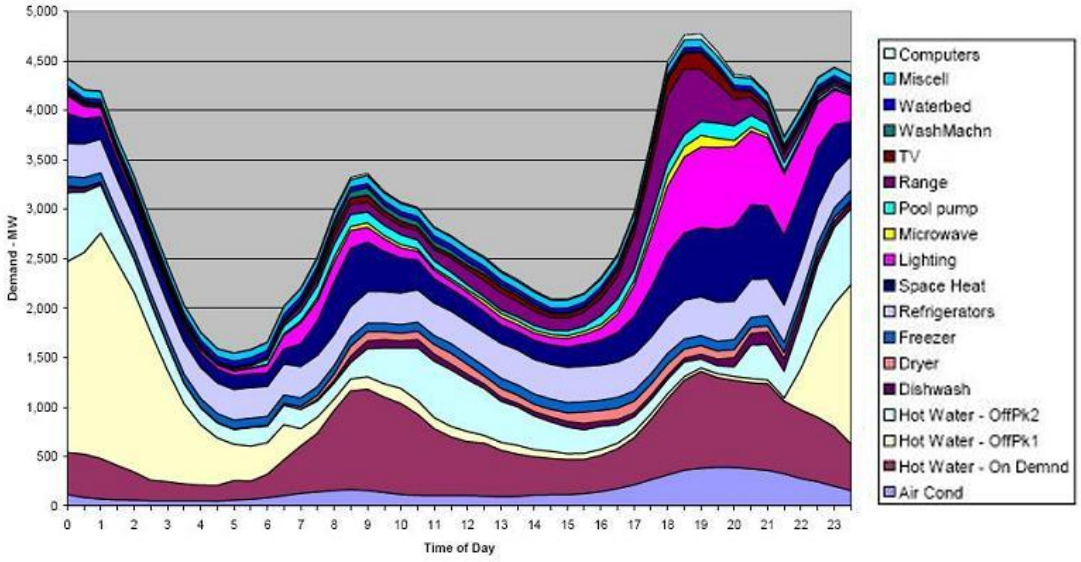

Figure 3 – Demand of electricity for various housing appliances throughout the day [58]

2.2.1 Agile Octopus Energy

Agile Octopus [13] will be used as the basis for the dynamic TOU tariff pricing. Agile Octopus uses 'agile pricing' to calculate new energy prices every half hour according to wholesale price and time. Equation (1) shows the equation used to calculate energy price, where 2.20 is the coefficient that includes distribution costs and is dependent on location, W is the cost of electricity for that period in p/kWh, P is peak time premium, and has a value of 12p from 4-7 pm. 33.33 is chosen to ensure the price is capped at 35p/kWh after VAT [62].

$$\min(2.20 * W + P, 33.33) \quad (1)$$

2.3 Time Series Forecasting

Time series data is a set of time stamped observations listed in time order. It is usually a continuous set of evenly spaced data, obtained through repeated measurements over time. Examples of time-series data can include stock market prices, energy consumption or temperature readings.

Time series forecasting is the process of predicting future data activity based on historical data. This is also known as autoregression. Forecasting is possible because of a few key components that make up time series data: autocorrelation, trend, stationarity, seasonality, and irregularity (Figure 4).

Autocorrelation – The similarity between the current and previous observations. The correlation with previous data is also known as 'lags'. When forecasting, the more lags are used, the more reliant the forecast is on past data.

Trend – The general direction (increase, decrease, neutral) of the time series, which can be either long or short-term.

Stationarity – Time series data is stationary if statistical properties such as the mean and variance of the data is constant over time. This is important because having these parameters constant makes the data predictable. If the data is non-stationary, then forecasting models would struggle to develop characteristics about past data, which would result in inaccurate forecasted values. Many models such as ARIMA (auto-regressive integrated moving average) are highly reliant in using past values, and therefore assumes the data it is using to be stationary [63][64].

Seasonality – A recurring pattern throughout the data that repeats at a fixed, known frequency. This pattern can repeat per year, month, minute and so on.

Irregularity – The short-term fluctuation or noise within the data and this is typically what decreases the accuracy of the forecast.

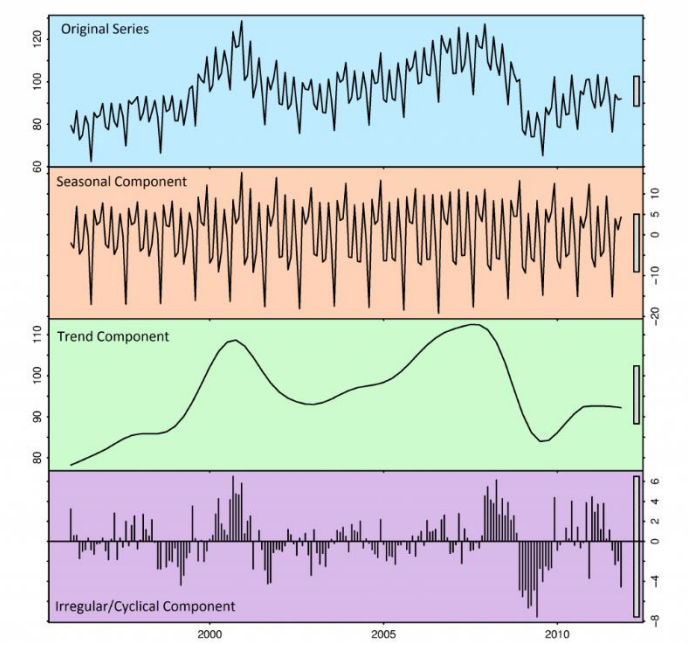

Figure 4 - Breakdown of the components that make up time series data [63]

2.3.1 Rolling Forecast

A rolling window in time series data, is the process of turning a single time series into multiple windows of fixed arbitrary size, with each new window shifting at least one time step further into the data throughout the full dataset. Figure 5 shows a single step rolling forecast, with a window size of 7, with 6 blocks used to train the data, and 1 block for forecasting. If more than one step is forecasted, then it would be a multi-step rolling forecast.

It is more advantageous for data that fluctuates a lot or is easily affected by external factors as the model will always accommodate for the most recent changes in data to perform calculations [65].

However, if the dataset is large, it can be computationally expensive and time-consuming as multiple forecasts are performed.

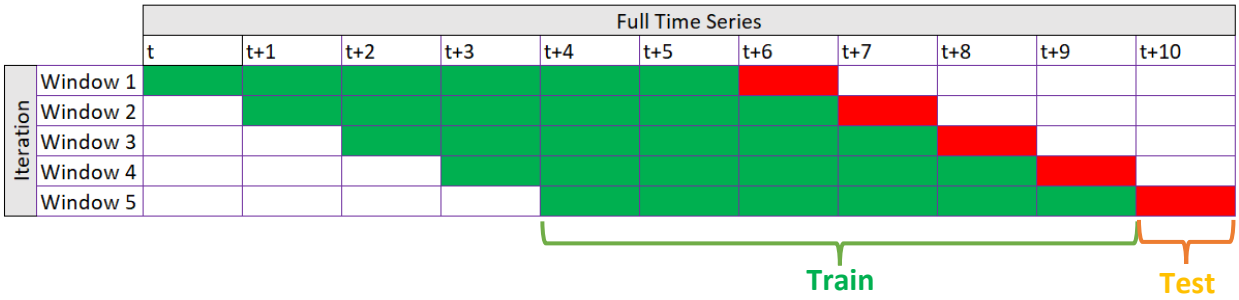

Figure 5 – Visualisation of a single step rolling forecast

2.3.2 Evaluation Metrics

To determine how successful our forecasts are, the accuracy can be measured using loss functions, also known as evaluation metrics. These metrics calculate the error of an observation, and each one indicates a certain aspect of error; therefore, multiple metrics are collated to determine if a forecast is acceptable.

Evaluation metrics can be categorised into two main areas: scale-dependent metrics and percentage error metrics. Scale-dependent metrics use the same scale as the data, the calculation is performed on. Percentage errors are unit-free making it useful if different datasets are compared against each other.

Metrics used to evaluate the forecasts in this report are:

Scale Dependent

- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)
- Mean Absolute Error (MAE)

Percentage Error

- Mean Absolute Percentage Error (MAPE)

More information is provided in Appendix 7.4.

3 Methodology

This section introduces the Recommendation Module, and describes the various methods used to develop it.

The Recommendation Module is a name given to the resulting product of this project which makes use of user drive cycle, TOU, and household load data to compute optimal charging time slots for the EV. It is made up of three main sections:

1. Forecasting Module
2. Classifier Module
3. Scheduling System

A flow chart of how the data and each module are linked can be seen in Figure 6. It is adapted from the Recommendation Module produced in a report by Shyang et al [42]. The main change is a swap in order between the classifier and forecasting module but the functionality of these modules still is still the same. The reason for this change will be explained in Section 3.5

The following section is written in operational order of the Recommendation Module, with Section 3.1-3.3 explaining how the input data (highlighted grey) is obtained. Section 3.4 talks about how the input data is forecasted. Section 3.5 is the classifier module which is used to determine periods when the user is not at home or driving, so it would not allocate charging periods during these times. Finally, Section 3.6 will demonstrate how the Recommendation Module determines and sets optimal charging periods for the user.

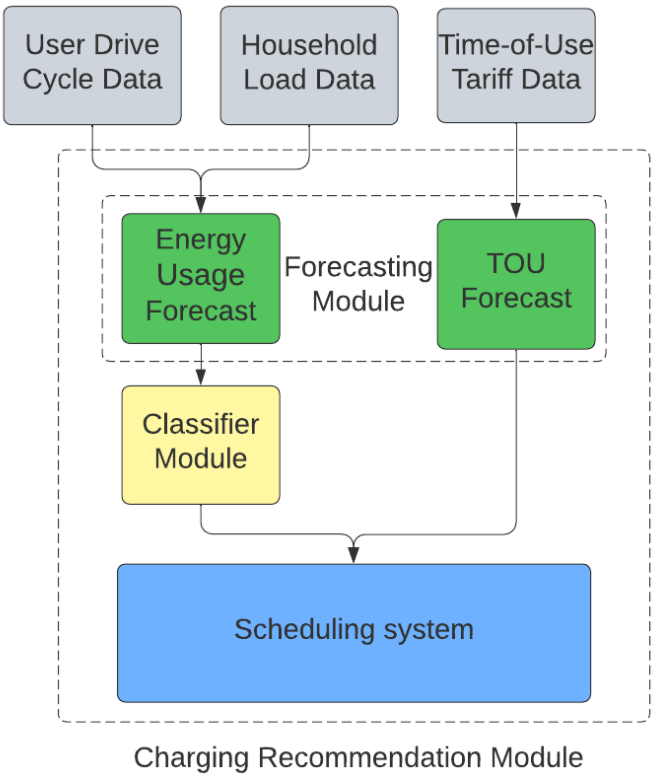

Figure 6 – The Recommendation Module flow chart

3.1 Obtaining EV data

As the project's main aim is to recommend ideal charging periods for the user based on the vehicle owner's driving routines, obtaining real EV user drive cycle data is key. To do this, data needs to be extracted from the vehicle. In this project, a few methods were attempted summarised below in 1 table below:

Table 1: Summary of methods used in this project to obtain EV data

Section	Method
3.1.1	Log CAN data using CAN bus data loggers
3.1.2	Finding datasets online
3.1.3	Pulling data via API

3.1.1 CAN Bus Data Logger – CANedge2

This section is a continuation of Section 5 of Shyang et al [42] where the setup procedure for obtaining vehicle data using a CAN bus logger was explained. In this project, we expand on their progress, in order to obtain SOC data and evaluate if this method is feasible for long term data recording.

CANedge2 setup procedure

CSS electronics' CANedge2 CAN bus data logger [66] was used to extract user drive cycle data from a BMW i3. The process of how to log the data is provided on the CSS website [67] and the high-level steps taken are shown in Figure 7. Firstly, the device was set up using a configuration file provided by CSS containing PIDs of certain vehicle parameters (Appendix 7.5). It was then plugged into the OBD2 port of a BMW i3 and left to log OBD2 data for an arbitrary period of time. The data recorded was saved onto an SD card which was then processed using DBC files into physical values. The full procedure of how the data was logged using the CANedge2 can be found in Appendix 7.6.

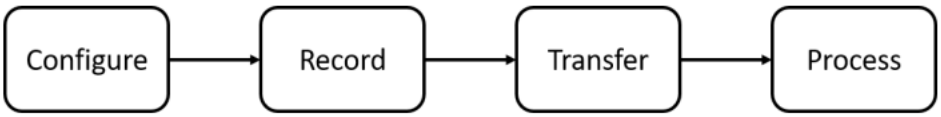

Figure 7- Overview procedure of how to use the CANedge2 device [67]

3.1.2 Online Datasets

Another method attempted was to find datasets online. After thoroughly searching the internet, one vehicle energy dataset (VED) was found [68][69]. The VED recorded real-time energy consumption data of 383 personal cars (unknown brand or model) from 2017 November to 2018 November in Ann Arbor, Michigan, USA. Of the 383 cars, three were EVs, representing less than 1% of all vehicles which is representative of vehicle ownership at the time, as the market share of EVs in 2017-2018 was around 0.6% [70]. Each vehicle had its own ID, with the EV IDs being 10, 455, 541. The data was logged using OBD2 data loggers, recording data such as battery SOC, vehicle speed, vehicle location and time, and then sent to a database via a cellular connection.

The data is saved in .csv file format in 1-week intervals. Therefore, to only extract the EV data, the CSV file was filtered by vehicle ID. The information needed was then copied onto a new CSV file. A selection of the data extracted can be seen below (Figure 8).

	A	B	C	D	E	F	G	H
1	DayNum	Timestamp(ms)	HV Battery Current[A]	HV Battery SOC[%]	HV Battery Voltage[V]	Latitude[deg]	Longitude[deg]	
2	4.698312648	0	-27.5	48.90244293	371	42.24393278	-83.74382583	
3	4.698312648	1100	-27.5	48.90244293	371	42.24393278	-83.74382583	
4	4.698312648	1600	22.5	48.90244293	376	42.24393278	-83.74382583	
5	4.698312648	2100	22.5	48.90244293	376	42.24393278	-83.74382583	
6	4.698312648	3200	22.5	48.90244293	376	42.24393278	-83.74382583	
7	4.698312648	4200	22.5	48.90244293	376	42.24393278	-83.74382583	
8	4.698312648	4900	22.5	48.90244293	376	42.24384194	-83.74302833	
9	4.698312648	5200	22.5	48.90244293	376	42.24384194	-83.74302833	
10	4.698312648	6300	22.5	48.90244293	376	42.24384194	-83.74302833	
11	4.698312648	6600	-22.5	48.90244293	371.5	42.24384194	-83.74302833	

Figure 8 – A sample of extracted user drive cycle data from the VED for vehicle id 455

DayNum represents the elapsed days since the starting reference date, i.e. DayNum = 1 = November, 1st, 2017, 00:00:00, DayNum 1.5 = November, 1st, 2017, 12:00:00 etc. The timestamp is in milliseconds and restarts every new journey taken.

Challenges faced with the data

From the data in Figure 8, we can see there are a few issues:

Firstly, DayNum remains constant and will remain constant during the journey, which means it is only showing the start time. This results in an incorrect plot, as a plot of DayNum vs Battery SOC (seen in Appendix 7.7) shows that the end SOC of one journey, tries to 'catch up' with the start SOC of the next.

Secondly, the SOC also remains constant for periods of time during a recorded journey, which could be due to the vehicle not moving, or the sampling rate of the OBD2 data logger not being in sync with the vehicle when requesting for data.

Lastly, it can be seen that the starting SOC of the current journey does not equal the end SOC of the previous, even if the journey taken is on the same day, meaning either the background systems of the car uses significant energy, the battery is leaking, parts of the journey are not recorded or data logging device doesn't record when vehicle is switched off.

However, these observations can be explained. In the VED, the coordinates of the vehicle are given. Therefore, for each journey, the start/end longitude, latitude coordinates were put into Google Maps to plot the route taken on that journey, to visualise if the vehicle is actually 'travelling', as well as to look for routines or common locations that the vehicle may travel to. A full table summary and a few routes can be found in Appendix 7.8.

From analysing the driver routes, a few observations could be made. It looked like the journeys were not recorded completely because often, the location of a trip stop/started in the middle of a junction, and even with journeys taken on the same day, there would regularly be a significant difference between the start and end location. However, the report that produced this VED [68] says that de-identification methods were implemented into the data so that confidential information such as where he/she lived or common locations visited were removed. These methods include random fogging, geo-fencing, and major intersections bounding. This

explains why there is missing data and that the results from the route tracing, match what the report said. This means the data is reliable for most of the journey but will not show full user drive cycle information.

Processing and Formatting the VED

To address the mentioned issues, the data must be formatted to ensure reliability of the VED, repeatability of the results as well as aids for a better understanding of the user drive cycle.

Timescale formatting

This is done in Excel by adding each row of the 'timestamp' during a single trip to the DayNum constant to obtain the real datetime. Time gaps between journey times are then forward filled until the next trip starts as neither DayNum nor timestamp is continuous throughout the entire dataset.

SOC data formatting

There are 2 scenarios experience by the SOC data:

1. SOC is lower at the start of the next journey

In this case, we assume the vehicle is not driven until the next trip and the data is forward filled using the last SOC value recorded. We will choose to ignore large drops in SOC since we do not know how the vehicle might have been driven before this period.

2. SOC is higher at the start of next journey

We want to reduce the discontinuity of the data so for this scenario, we assume the user charges their vehicle at some point before the next journey. Currently, the increase in SOC is instantaneous since the period that the EV is charged is not recorded. However, to be more realistic and representative of real EV charging, the charging period is simulated.

To determine the approximate time it takes for the EV to charge, a few key assumptions need to be made:

1. Assume all charging occurs at home on a dedicated 7kW home charger [71][72] as surveys have shown that 80% of people only charge at home in the UK and currently, 7kW charging walls are becoming much more popular [73].
2. From collated journeys summarising the reduction in SOC and distance travelled (Appendix 7.9), it can be seen that the car has a max predicted range of approximately 152 miles, and an average predicted range of 80 miles. Furthermore, the data was recorded from 2017-2018 [69]. From these empirical observations, we can assume the vehicle is a 30kWh Nissan Leaf as these statistics match officially tested data [74] and in 2017, the only vehicle matching these stats were the 30kWh Nissan Leaf [75].
3. We will assume that the user will charge their vehicle immediately after the journey ends. This is to ensure repeatability many EV owners charge immediately after they get home from work [76].
4. The final assumption we will make is that the charging rate for the EV is constant. As shown in Figure 9 which shows a plot of SOC rate vs SOC percentage for 30kWh Nissan Leaf [77], we can see that the rate of charge up to 80% is almost constant, until it drops off at around 80% SOC. This is consistent with Recommendation Modules plan

to maintain SOC percentage between 20%-80% to prolong battery life [78]. Therefore, charging a 30kWh EV with a 7kW home charger (limited to 6.6kW due to Nissan Leaf's onboard AC charger [74], it would take approximately 4 hours 30 minutes to charge from 0-100%, matching official stats [79] which says it should take approximately 4-5 hours giving convincing evidence that this is the same car.

Considering all these assumptions, we can finally calculate and set a charging rate for the EV:

$$Charging\ 30kWh\ battery\ from\ 0-100\%\ at\ 6.6\ kW:$$

$$\frac{30}{6.6} = 4.54 = 4\ hours\ 32\ minutes$$

$$1\%\ of\ charge\ will\ take\ 0.0454 = 2\ minutes\ 44\ seconds$$

$$\therefore Rate\ of\ charge = 0.0061\ percent/second$$

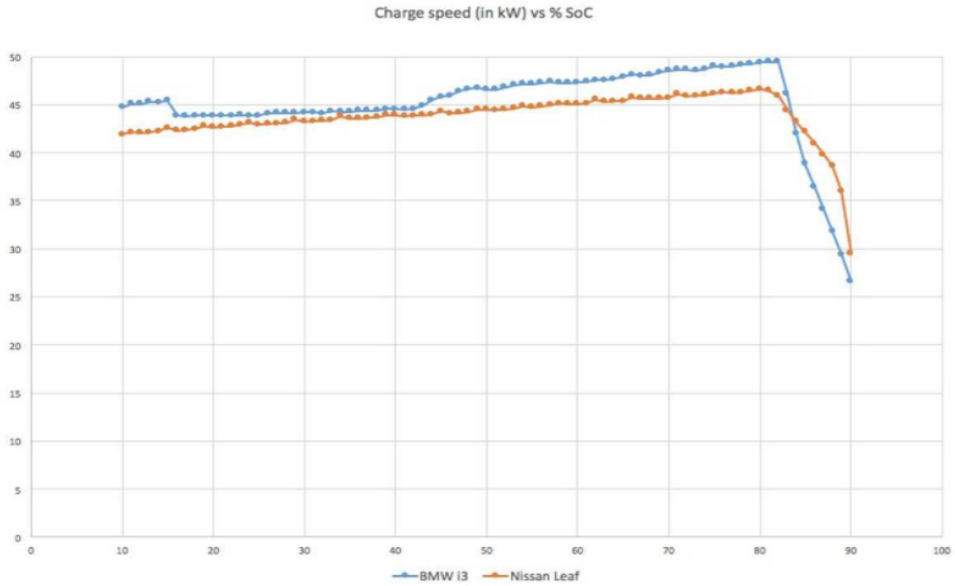

Figure 9 - Rate of charging for a 30kWh Nissan Leaf at different battery percentages [77]

3.1.3 Obtaining Data via API

A method attempted later in the project which showed potential in obtaining EV data is using APIs. As we know, auto manufacturers use APIs to collect data about their cars, many people have also attempted to retrieve this data from their own cars. We will focus on the Tesla API. Typically, APIs are not meant for public use. However, enthusiasts have been able to reverse engineer endpoints through Tesla's mobile application and vehicle software [80].

There exist a few unofficial Tesla APIs created by enthusiasts to allow Tesla owners to access more information about their vehicle [80][81]. These two APIs in particular used OAuth, which is the old authentication protocol. However, recently Tesla updated its authentication system, deprecating (no longer uses) the OAuth endpoint in favour of using OAuth2, which acts as a completely new protocol [82]. This meant that the procedure for retrieving the authentication

token needed to be rewritten. An API documentation, which is still in working progress [82] provides the only information about this update online.

An implementation of the API can be found on GitHub [83], where the API was further extended by implementing a user interface to visualise the data. In this experiment, access to a Tesla vehicle was provided by our industrial partner. The full procedure of how this API is set up can be found in Appendix 7.10.

3.2 Obtaining Household Load Data

To minimize the impact on the grid, this project makes use of household load profiles and TOU tariffs to recommend charging periods that benefits the grid as well as the user. Integrating household load data into the recommendation algorithm is a novel approach first introduced by Shyang et al [42]. Taking into account household load provides key information, such as when the user is at home and how much energy is used, so we can schedule EV charging in a way that also minimises household load on the grid.

Household electricity load online dataset

The data used was an electric load dataset of UK households [84]. Measurements were taken in 8-second intervals from 20 different houses over a period of 2 years for 9 specific appliances. The data we will focus on is the household aggregate, which is a sum of all the power used by the appliances at each instance.

The dataset [85] also mentioned that houses 3,11,21 had solar panels installed that caused interference in the reading: “Solar panels appeared as additional power consumption resulting in a bell-curve-shaped power consumption increase during the day with significant noise due to weather changes, such as clouds.” Therefore, the data from these houses will be avoided. All other houses had reliable data and so for this report, a house number will be chosen arbitrarily. We used house number 4.

3.3 Obtaining TOU Tariff Data

Time of use (TOU) tariff data is a hugely important set of data used in smart charging algorithms as it not only provides information about the cheapest times for electricity throughout the day, but also inadvertently indicates the condition of the grid (peak/off-peak). For this project, we use dynamic TOU tariff as mentioned in Section 2.2, since it provides the highest opportunity of rewards when charging EVs.

Price data was found from website [86] which pulled Agile Octopus Energy tariff data using the Octopus Energy API. Datasets come in 14 regions, and there is data for Agile import tariff data as well as output tariff . We will focus on Agile import as it is the price we pay for buying electricity. Each day Octopus Energy releases new tariff pricing for the upcoming day so the tariff plot is up to date. The data starts on the 18th of February 2018 up to the current date (08/01/2022).

3.4 Forecasting Module

The forecasting module is made up of 2 parts (Figure 10), the energy usage and TOU forecaster. The energy usage forecaster takes inputs from both the household load and the EV data previously obtained to forecast, energy consumption of the user, and to build up user behaviour characteristics since was previously mentioned, the smart charger's aim is to suit the user's lifestyle.

The TOU forecaster is similar to what Octopus Energy does when they set energy pricing for the next day, however this project aims to forecast at least one week in advance so that the user can take advantage of the cheapest charging costs in the long-term future.

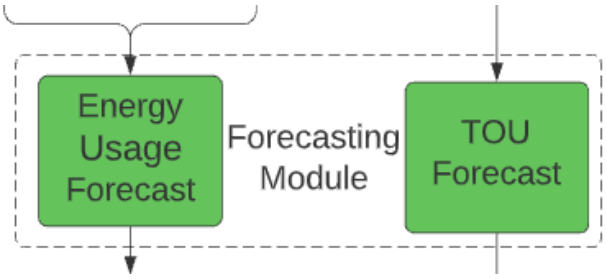

Figure 10 – Snapshot of EV forecasting module from the Recommendation Module

Forecasting tools and requirements

Time series forecasting experiments were done remotely using Myriad, a high throughput general-purpose computer provided by UCL Research Computing. The process of submitting jobscripts to Myriad can be found in the Appendix 7.11.

Main libraries used are *sci-kit learn, fbprophet and statsmodels.* Datasets are the ones previously mentioned. They are collated in the GitHub repository for this project [87].

In this paper, we will explore time-series forecasting of TOU data, to forecast future energy prices because at the current stage, this data is most complete.

For statistical evaluation of the forecasting models, metrics used will include: Mean squared error (MSE), Root mean squared error RMSE, Mean absolute error MAE, Mean absolute percentage error MAPE which are explained in Appendix 7.4.

3.5 Classifier Module

In Shyang et al [42], the classifier module was placed before the energy forecasting module. This did not make sense because the classifier module has two functions:

1. Determining future time periods when the user is at home

The classifier module determines when the user is not at home using a threshold. The threshold is based on empirical observations which is the average power drawn from the house when no one is at home/they are sleeping. If it goes above this threshold, then the classifier module will indicate that they are at home. We will make the assumption that the plots produced are of a

40 hour a week day-shift worker's lifestyle [88][89]. We also assume that from 12midnight to 7am they are at home [90].

2. Determining future time periods when the user is not driving their vehicle

The classifier module can deduce when the user is not driving their vehicle by scanning the SOC profile and looking for negative changes in SOC (negative gradient). This helps ensure the scheduling system (Section 3.6) does not allocate charging periods during these times. If the module was placed before the forecasted data, then it would only be useful in determining these time periods for past data.

The classifier module takes the forecasted household load and user drive cycle data as inputs. To visualise how this module functions, a 2 day period is analysed. Since datasets all come from different sources, this means the year of which the data was recorded is also different, however to try to maintain consistency of results, the same month and date from both datasets are used: 9/11-16/11.

3.6 Scheduling System

The last stage of the Recommendation Module is the scheduling system. Inputs are from the classifier module and the TOU tariff forecast directly (Figure 6). After the determining the location of the user as well as forecasting the future energy consumption and prices, the system takes these into account and follows an algorithm shown in the flow chart (Figure 11) to locate the optimal charging periods that are available for the user and have cheapest energy costs in the closest time period.

Battery health is also considered in the smart charger, where the lower limit is the sum of the lower battery buffer (default set to be 20%) plus emergency reserves which is used to reach closest charging stations if battery is critically low (default is 0%). An upper battery buffer is default set at 80%. The default values are set because research and experiments have shown that maintaining battery SOC between 20-80% can preserve battery health and prolong the battery's max capacity [78].

The algorithm works as follows:

When the scheduler module first begins, it calculates if the current EV SOC is greater than the next day's SOC consumption plus battery buffer/emergency reserves.

If the SOC is sufficient for next day's journeys, then it will check if it is greater than the upper limit of the battery which if true, will end the algorithm to prevent battery degradation. Otherwise, it will charge up to this limit at cheapest charging slots available to not only lessen the worry of range anxiety but also take away excess energy from the grid, supporting the cycle of supply and demand of the grid.

If SOC is less than the required, the main concern would be if SOC is less than the lower limit which if true, results in allocating a cheap charging period in the near future to maintain battery health and ensure the car is not at risk of falling to critical battery SOC due to vampire drain (see Section 4.1.3 for more information).

Most users charge their vehicle at home, however if a situation arises where they are not able to, then not only looking for charging period but also charging location is also an important function.

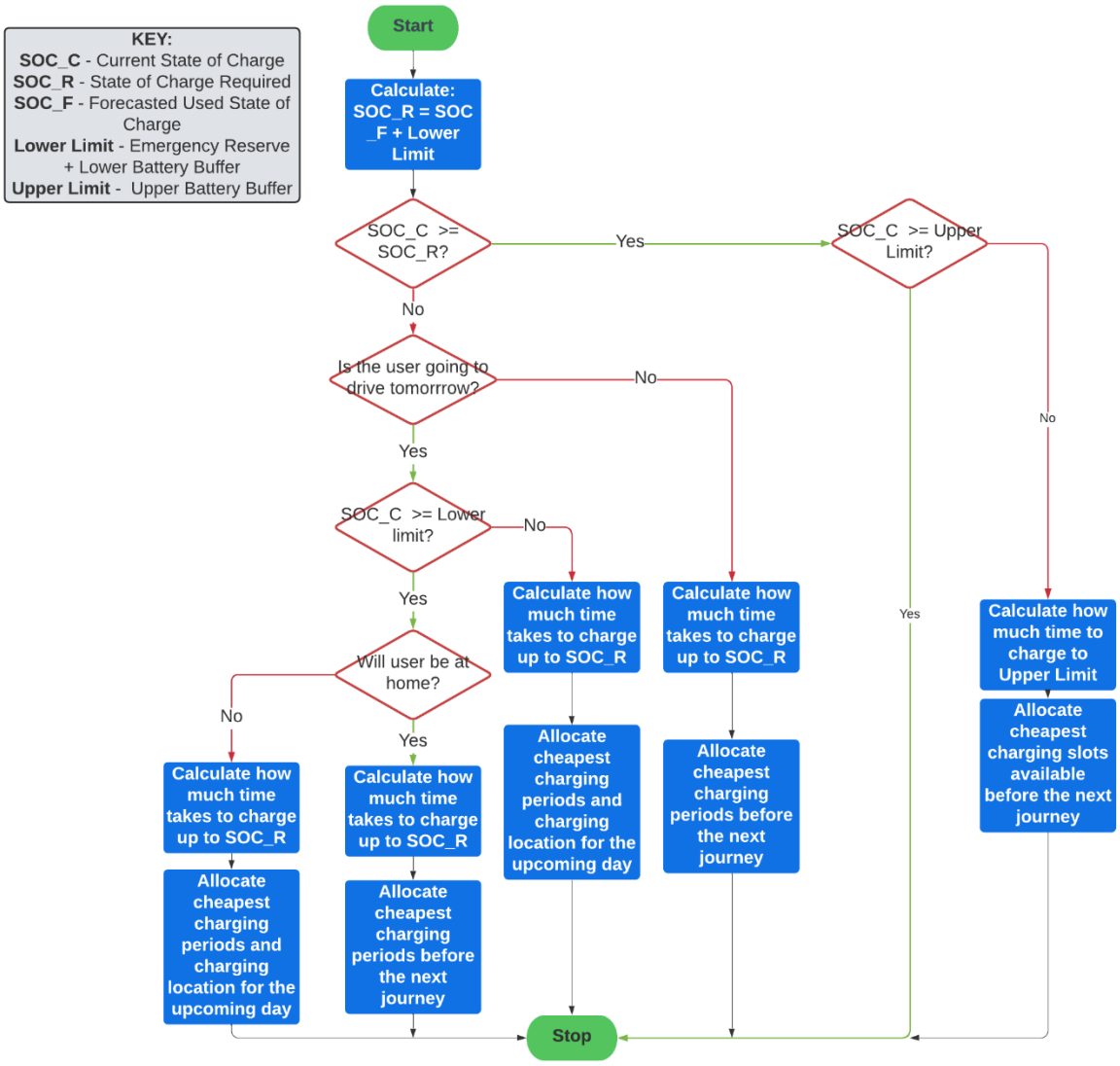

Figure 11 – Flow chart algorithm of the Scheduling System for the Recommendation Module

4 Results and Discussion

This section describes and discusses the results achieved from the methods used in the previous section as well as results of each individual module.

4.1 Evaluating The Methods Used of Obtaining EV Data

This section presents the results of the different methods to extract EV data and summarises the advantages and disadvantages of them.

4.1.1 CAN Bus Data Logger

Data is logged from the CANedge2 into a new folder each time the car is turned on. The data logs continuously, requesting and recording data every 2 seconds. Once the raw data is extracted, it is then processed into a human-readable format, shown in Figure 12 which shows a snippet of data for vehicle speed. This particular set of data was recorded continuously for 2 hours and 34 minutes.

TimeStamp	CAN ID	Signal	Raw Value	Physical Value
2021-03-23 09:35:42.251350+00:00	2024	S1_PID_0D_VehicleSpeed	0	0
2021-03-23 09:35:42.751350+00:00	2024	S1_PID_0D_VehicleSpeed	0	0
2021-03-23 09:35:43.251350+00:00	2024	S1_PID_0D_VehicleSpeed	0	0
2021-03-23 09:35:43.752050+00:00	2024	S1_PID_0D_VehicleSpeed	0	0
2021-03-23 09:35:44.252100+00:00	2024	S1_PID_0D_VehicleSpeed	0	0
2021-03-23 09:35:44.752000+00:00	2024	S1_PID_0D_VehicleSpeed	0	0

Figure 12 – A sample of extracted data using the CANedge2 device which showing vehicle speed

A visualisation of this data for a 12-minute period can be seen in Figure 13 and the longer 2.5 hour period can be seen in the Appendix 7.12.

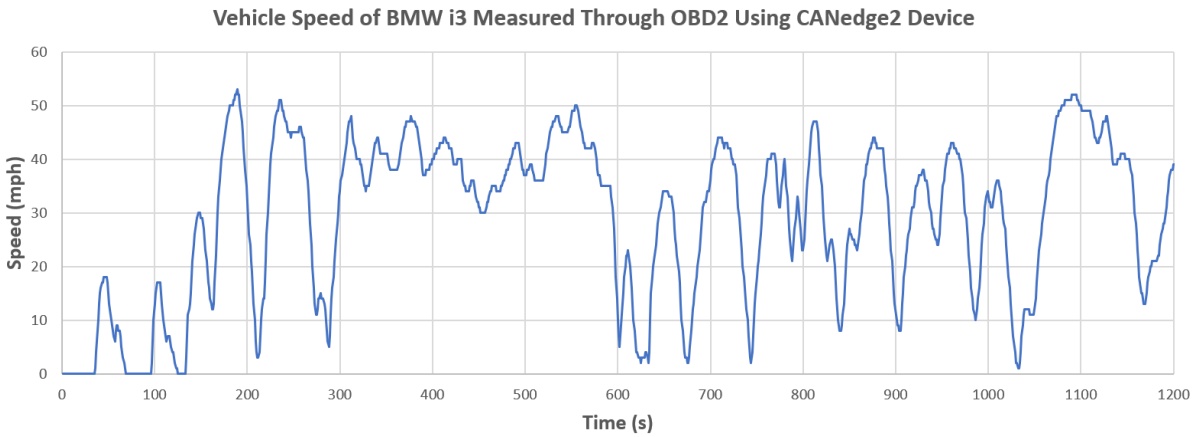

Figure 13 – The extracted data using the CANedge2, plotted showing vehicle speed vs time

After the vehicle data (speed, engine rpm, engine temperature) was obtained using the config file provided by CSS, extraction of SOC data was attempted. The PID used was found on a BMW enthusiast forum [91]. However, when uploaded onto the config file, no data was logged. This suggests that either the PID is incorrect, or the DBC file used to convert raw CAN data into a human-readable format was also incorrect. Unfortunately, the SOC PID for BMW i3 is very difficult to find as car manufacturers tend not to release this information to the public and

there is minimal information online, which means we cannot confirm whether the PID we have is correct.

Furthermore, inserting a logger into OBD2 port can occasionally trigger alarm warnings from the BMW i3 in certain situations for example, when the device is initially plugged in, or if the device is plugged in when the vehicle is locked [92]. Therefore, if this method were to be used, the driver may need to remember to plug and unplug the CANedge for each journey, which may become inconvenient.

After successfully logging vehicle data using a CAN bus data logger, it is evident that this is a feasible method for logging user drive cycle data. The "record, transfer, process" process is now well-documented and therefore, once the correct PIDs are input into the config file, it should be possible to record user drive cycle data.

4.1.2 Online Datasets

Following the data processing process, a plot of DayNum vs Battery SOC with correct formatting can be seen below for a period of 10 days (Figure 14).

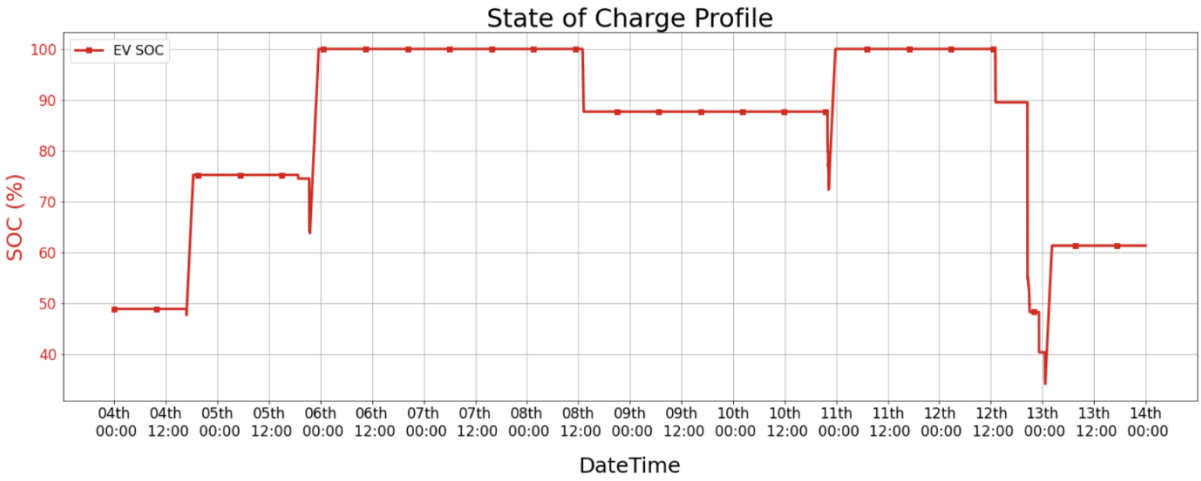

Figure 14 - SOC profile produced from the VED for one month with correct time scale and forward filled data

From the plot, we can see that most journeys taken were in the evening and that a few times in the 10-day period, the user charged their vehicle up to 100% even though it was not necessary for the journeys they were going to take the following day.

Because a considerable number of assumptions are made based on empirical observations and surveys, this means the SOC profile is not entirely representative of the real data. However, in general, it gives a good idea of how the user drives and charges their vehicle, and the profile is a good baseline to compare once better methods of obtaining EV data are used.

The public vehicle data obtained from this dataset contained sensitive information (longitude, latitude coordinates), which was why de-identification methods are used. Coordinate data is incredibly useful, and we can potentially take inspiration from this dataset to also record vehicle

coordinates on our private dataset, as it makes determining common locations where the user drives to, such as home coordinates, much more accurate.

4.1.3 API

Figure 15 shows a few instances of live Tesla data captured in 2-minute intervals using the Tesla API. Data that was requested were: battery SOC, estimated vehicle range and the charging state. This is a significant milestone in the project as it is the first time SOC data has been captured from an EV.

Unfortunately, because of time constraints, finding a method of logging this data long term was not achieved. One reason for this is because while we were initially trying to obtain instances of data, we discovered that the data can only be requested if the vehicle is awake/running. Applying commands to wake the vehicle were attempted but it resulted in the API crashing.

If data from a Tesla were to be logged continuously, this would mean the Tesla is constantly kept awake, and therefore it will be important to consider an EV characteristic called 'vampire drain'. This occurs when the Tesla is on stand-by mode instead of sleeping, which means that overtime, the battery can decrease, typically draining SOC by 1-5 % per 24 hour [93][94]. If current SOC is low (less than 10%), this additional loss can cause serious risk of damage to the battery so this process should be exercised with precaution.

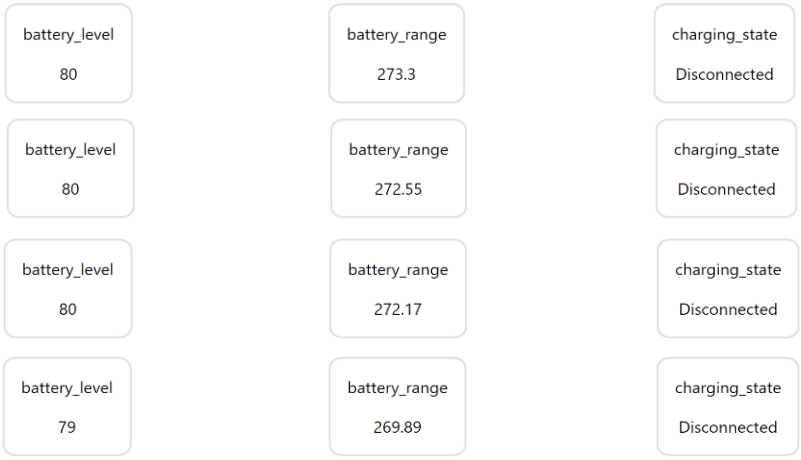

Figure 15 – Instances of SOC and other EV data captured using Tesla API

This new approach to obtaining EV SOC data has proven to be partially successful. Furthermore, the data received is automatically in a human-readable format so it does not need to be processed afterwards. However, drawbacks of this method are that the initial learning curve to understand and implement the API is relatively difficult. Secondly, this method may be difficult to scale because making direct request to the vehicle requires an authentication token from the API, which in turn requires sensitive data, such as the vehicle owner's email and password.

4.1.4 Obtaining EV Data Summary

Key details from the methods used to obtain SOC data in this project is summarised in Table 2.

The most recent method attempted of obtaining data through APIs shows the greatest potential over other methods, as it allows complete remote access and flexibility in the data that can be recorded.

Public datasets that log EV data are currently difficult to find. However, over time as the ownership percentage of EVs increase, these datasets will be more abundant as well. If logged data from these datasets do not contain sensitive information which results in data-fogging, it may in fact be hugely advantageous as these datasets usually log data over hundreds of vehicles

Lastly, the CAN bus data logger method has shown that it can log user drive cycle data over time, and once it is set up it is as simple as 'plug and play'. However, obtaining PIDs for specific vehicles, especially new EVs is difficult and so unless reverse engineering of this data is attempted, or more information regarding this can be found online, then for the moment work on this method can be paused.

Table 2: Summary of the pros and cons of different methods to obtain EV data

Method	Pros	Cons
CAN bus data logger	- Can be left alone after setup - Data be logged to the cloud	- Specific vehicle information is difficult to find - Setup costs are high - Initial setup needs to be in person - Can occasionally set off vehicle alarm - Requires decoding after obtaining data
Finding Online Datasets	- Quickest and easiest to obtain	- May require a lot of data pre-processing before it can be used for anything else - Not entirely representative of full user behaviour due to the number of necessary assumptions made because of incomplete data
API	- Can be done entirely remotely - Can be left alone after setup	- Steep learning curve - Requires a laptop to run in the background - Requires sensitive data (email and password) - Amplifies the phenomena of vampire drain

4.2 Household Load Data Results

Since data was recorded so frequently, there is a substantial amount of data which when plotted, not only makes it difficult to read (Appendix 7.13), but is also unnecessary since we only need to use this data to determine an approximate time when the user is at home. Therefore, the data was resampled, using data from every 50th row in the dataset (using every 5 minutes) to visualise the graph more clearly and to maintain the general characteristics of the plot.

To reduce the impact of noise and fluctuations in the data, a moving average was also plotted (orange line). This takes the arithmetic mean of the last 'n' values, hence averaging out any outliers and also makes the trend much more visible. In Figure 16 below, the moving average was taken for every 250 rows or 30 mins of data as this value represented the overall trend well.

The resultant aggregate load profile of house number 4 for 1 day is plotted below in Figure 16. And below that, Figure 17 is the load profile for individual appliances.

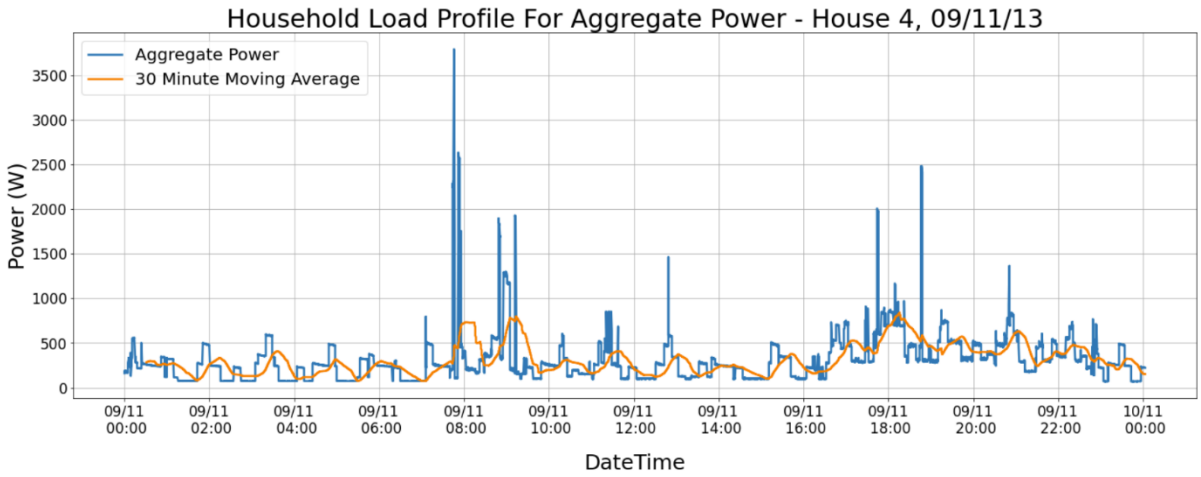

Figure 16 – House number 4's load profile of aggregate power for 1 day, showing both sample data (blue) line as well as a 30 minute moving average (orange)

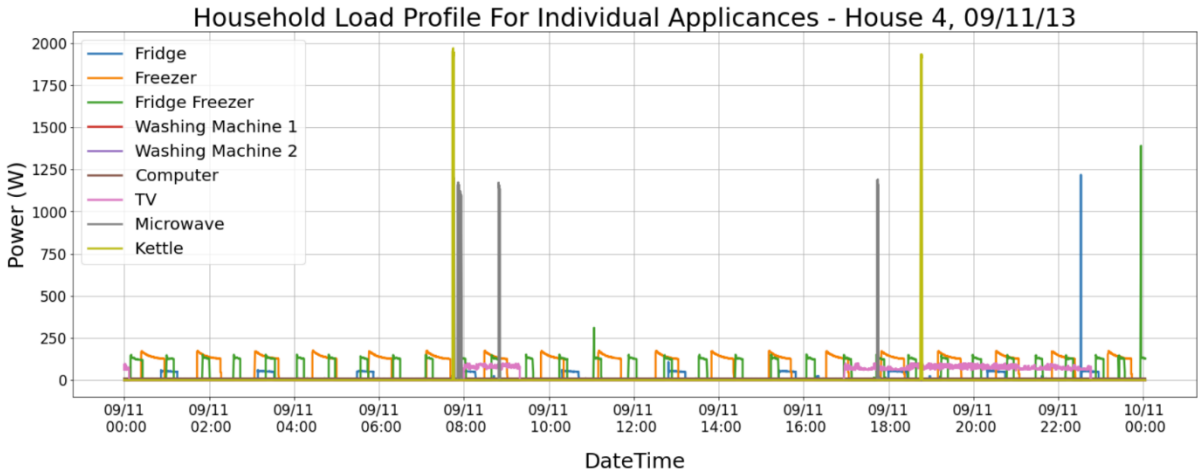

Figure 17 - House number 4's load profile of individual appliance power consumptions for 1 day. The sum at each instance of time will produce the plot of the aggregate power in figure 16

From these 2 figures, we can see why the aggregate plot spikes so quickly at various times of the day, for example 7:40am in the morning – because two high power household appliances were used at the same time (kettle and microwave). We can also start to deduce from the moving average a threshold power for when the house is "dormant" i.e. when the user is not at home or sleeping, which is essential information for the classifier module.

Household load data for this project is now well understood and documented. After processing and analysing the data from house number 4, we can determine that this dataset [85] is reliable and can be used for future house energy related work.

Including this type of data will benefit the project greatly as we will be able to predict the user's living behaviour more accurately throughout the day. However, it is necessary to add that this dataset did not come from the same person or location as the EV dataset, and therefore may affect the accuracy of the recommended optimal charging period if both users lived vastly different lifestyles.

4.3 TOU Tariff Data Results

A plot for the entire dataset up to the current date taken is shown in Figure 18. From this plot we can see that the price can fluctuate from a maximum cap set by Octopus Energy of 35 p/kWh to a minimum of -10 p/kWh where the user would actually be paid to use energy. Thus showing that that the consumer should pay attention to when they use energy to capitalise on the benefits of dynamic tariffs.

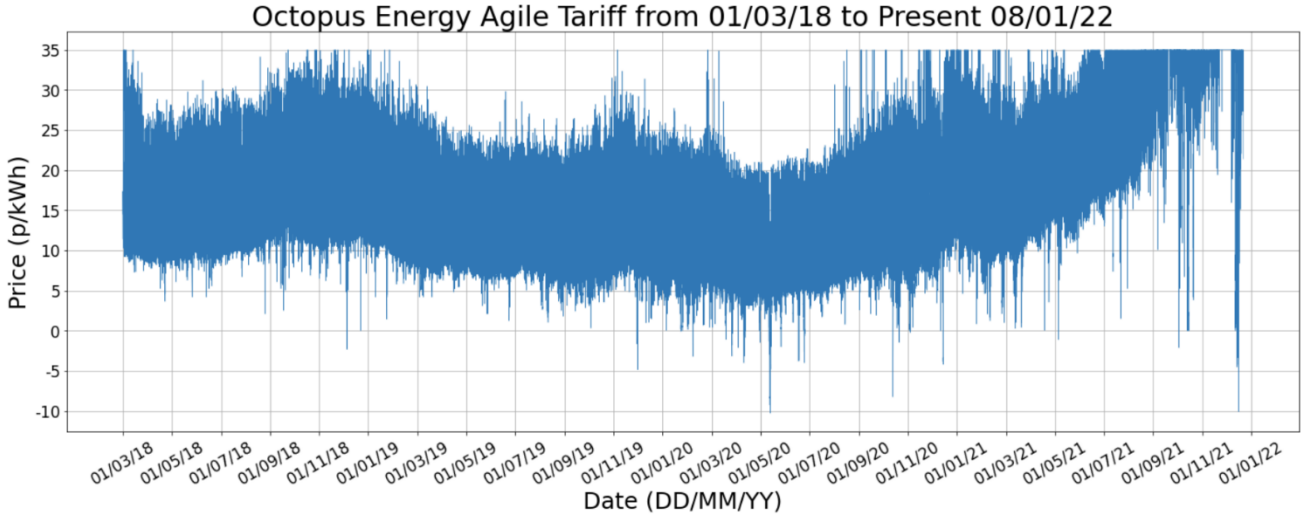

Figure 18 – Plot of Agile Octopus prices from 1st March 2018 to 1st December 2021, location – London. Plot shows the price caps at 35 p/kWh as well as the varying trend for energy price throughout the year

An important observation of the graph is the trend in energy prices which vary depending on the season, for example, from 01/03/18 onwards, the price for energy increases up to the winter months, peaking around 01/11/18 before the graph trend decreases back down for spring season. This is expected since during winter months, people may turn on the heating for extended periods of time and daylight time is shorter so lights will be kept on for longer. As EV ownership increases over time, this trend may become even more significant because EV

driving range is highly affected by cold temperatures which can hugely reduce the battery performance as car functions, such as heating or defrosting is used more often, resulting in the need for more frequent charging [95].

However, this seasonal trend cannot explain the sudden increase in energy prices starting from around 01/08/20. Dynamic TOU tariff prices can be affected by external factors such as weather, wholesale costs and energy distributor conditions, which is why from 24 August 2020 onwards, Octopus Energy reported several times that renewable energy generation was low. The weather was unusually cold and a few power stations were offline [96], hence the increase in energy prices. It is these particular anomalies in the TOU tariff price that the forecasting module needs to pick up on to take advantage of dynamic TOU tariffs.

To analyse in more detail the daily fluctuations in prices, a plot for an arbitrarily chosen week of TOU tariff data in November 2018 was generated. Looking at Figure 19 a clear seasonal pattern with 2 significant features can be seen. The first is that at 4-7pm each day, the price climbs rapidly, which is explained by the equation in Section 2.2.1. The second feature is, that the cheapest price of energy occurs at around 2-4am each day, and therefore the goal of the scheduling system would be to determine which day over a certain period has the cheapest energy.

Also, it is interesting to note that Octopus Energy does not distinguish between weekends and weekdays, because energy profiles during weekends may have a flatter profile, without sudden peaks like the weekday. If the user were to use Octopus Energy, this shouldn't affect how optimal EV charging periods are identified. However, it could allow for a wider range of charging periods.

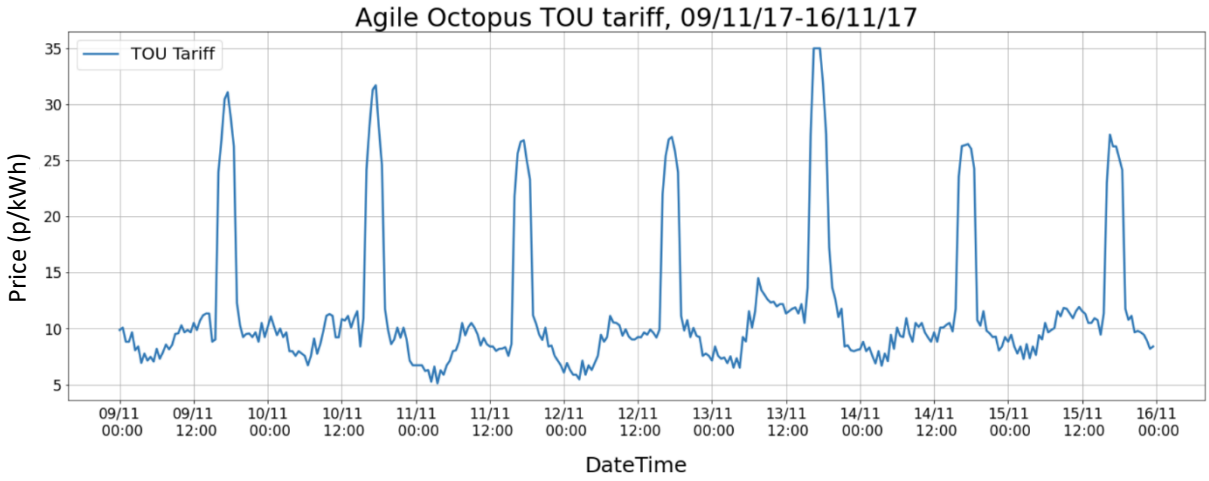

Figure 19 – Plot of Octopus Agile price for 1 week in November 2017, location – London. A shorter time period of data shows how the price fluctuates according to different times of the day

After acquiring dynamic TOU data, we have seen that the graph has long and short-term seasonal trends. Furthermore, when energy prices are stable, it has shown greatest potential in reducing impact of EV charging on the grid by incentivising cheaper energy costs at off-peak times. However, as was shown, the price of energy can fluctuate rapidly due to external factors, which can result in higher-than-average energy prices or occasionally paying the consumer to take excess energy off the grid. For instance, currently Octopus Energy advises not to use

dynamic TOU tariff because energy prices are not only at an all-time high, but also extremely volatile [96].

4.4 Forecasting Module

This section presents the results of forecasting TOU data using various time series models. The models are evaluated using metrics previously mentioned in Section 2.3.2.

4.4.1 Baseline Models

Baseline forecasts produce a basic machine learning forecast that is typically used as a comparison with more complex models to know how well perform.

Baseline models are fast to implement, simple in its calculations (no intelligence) and repeatable [97].

In this project, 3 baseline models were used: Mean, Naïve and Seasonal Naive. The mean baseline model calculates the average price for all the test data and this value becomes the forecast. Naïve model forecasts the last value in the test data as a flat-line forecast. Seasonal Naïve forecast, forecasts the last seasonal period of test data and repeats the seasonality for the rest of the test data. More information of these baseline models can be found in the Appendix 7.14.

As can be seen in Figure 20, the seasonal naïve forecast replicates the seasonality of the real TOU data well. This is because on most days, the TOU price fluctuation is stable and repetitive which is where seasonal naïve works best. The other 2 baseline models perform flat-line forecasts. Table 5 shows that naïve model has lower errors than the mean, this is likely due to the surge in price for peak-times that greatly affect the average price. And as expected, the seasonal naïve outperforms both other baseline forecasts.

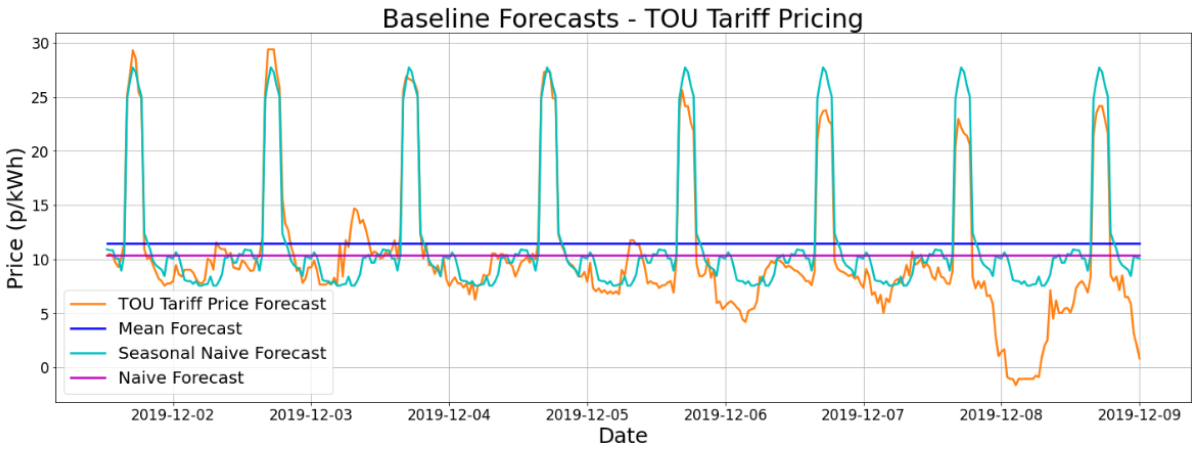

Figure 20 – Baseline model forecast for TOU tariff data. 3 baseline models are plot (Cyan – Seasonal naive, Pink – Naïve, Blue - Average) and compared to the real tariff data (Orange)

4.4.2 ARIMA/SARIMA

For ARIMA/SARIMA models, selecting hyperparameters: autoregressive, differencing moving average and seasonality (SARIMA) parameters (p,q,d) is key to achieve accurate

forecasting performance. As such, hyperparameter tuning Is necessary to determine which values would work best with this data.

Firstly the data needs to be checked for stationarity, which is done using a rolling statistics test to visualise if the data is stationary, and an Augmented Dickey Fuller test (ADF) to statistically determine it. If the data isn't stationary, differencing would be needed. Once stationary data is confirmed, the 2 other hyperparameters p,q can be found. 2 methods to find these values are used: Autocorrelation/Partial Autocorrelation (ACF/PCF) plots and Akaike Information Criterion (AIC). Further information on how these tests are used can be found in Appendix 7.15. The hyperparameters used are summarized in Table 3.

Table 3: ARIMA/SARIMA optimal hyperparameters for TOU forecasting

	Trend (p,d,q)	**Seasonal (P,D,Q,s)**
Optimal values	(4,0,5)	(4,0,5,48)

The plots for ARIMA and SARIMA for a 7 day period is shown in Figure 21. ARIMA (cyan line), produces a flat forecast similar to the naïve and mean baseline models.

SARIMA demonstrates a good forecast for seasonality however it doesn't follow the trend of peak time prices well, as it maintains constant peak price throughout the test data. Furthermore, it is unable to detect the outlier data that occurs on 12/08/19.

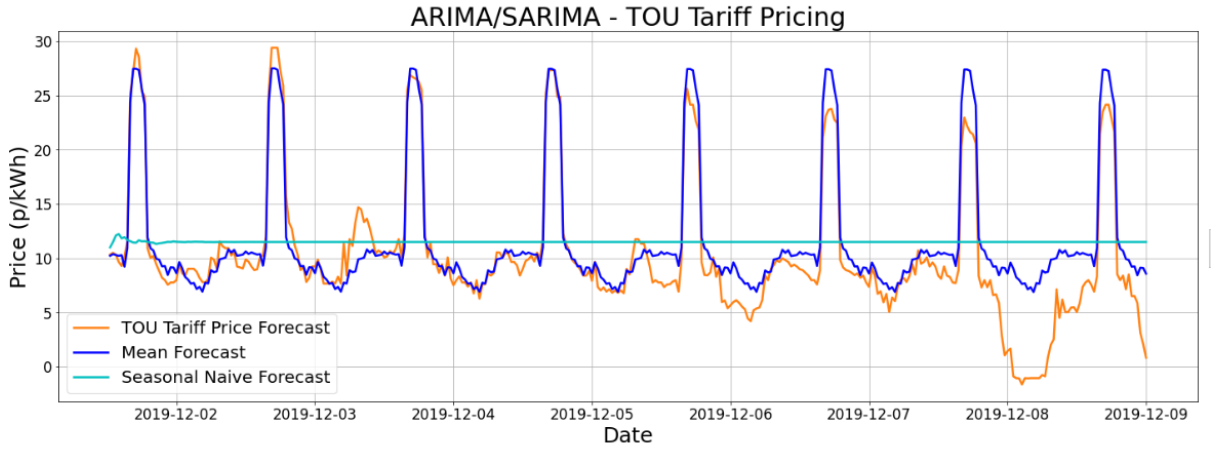

Figure 21 – ARIMA (Cyan), SARIMA (Blue) plot for TOU tariff data compared to real tariff data (orange)

4.4.3 FBProphet

FBProphet is a time series forecasting model developed by Facebook. It is an additive model that can fit periodic data, more information about the mathematics behind it can be found in Appendix 7.16. Like ARIMA, it too also has tuneable hyperparameters to improve forecasting accuracy. Cross validation is performed [87] to evaluate the different combination of parameters to see which was optimal. Results are summarised in Table 4.

Table 4: FBProphet optimal hyperparameters for TOU forecasting

	Change_prior_scale	seasonality_prior_scale	daily_seasonality	weekly_seasonality
Optimal values	0.5	1	48	7

Figure 22 shows the plot using Prophet. Similar to SARIMA, it also detects seasonality well. It is slightly better than SARIMA at detecting the trend as the peak price does fluctuate, but it is not significant enough to match the real data. It also was unable to the outlier data that occurred on 12/08/19.

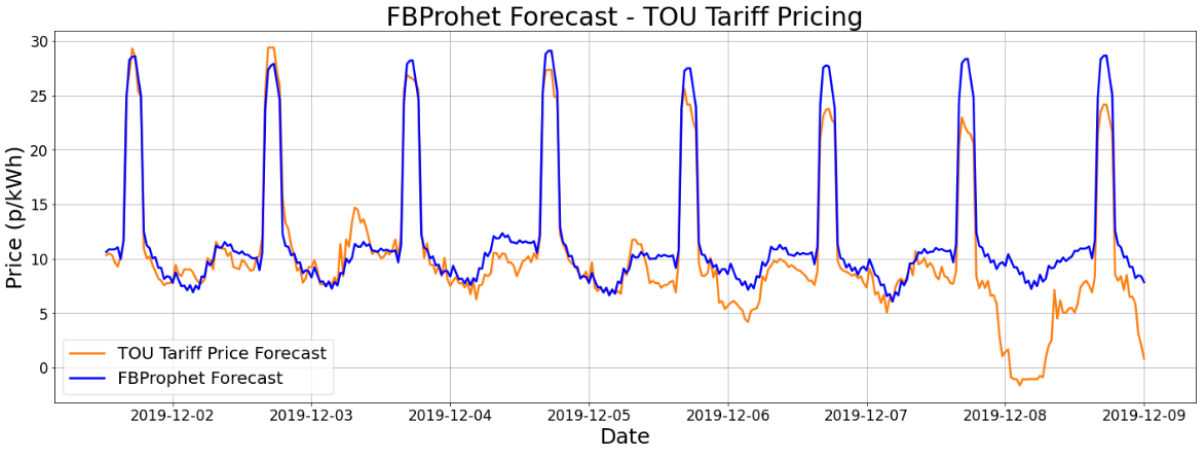

Figure 22 – FBProphet (Blue) plot for TOU tariff data compared to real tariff data (orange)

4.4.4 Rolling ARIMA

Rolling forecasts are a popular method used alongside ARIMA models. The data was trained and fitted to the model and a 1 step rolling forecast use performed where the model forecasted 1 step into the future.

Figure 23 shows the results of the rolling ARIMA forecast. The forecast (blue line) follows the trend well and fluctuates very similar to the real data.

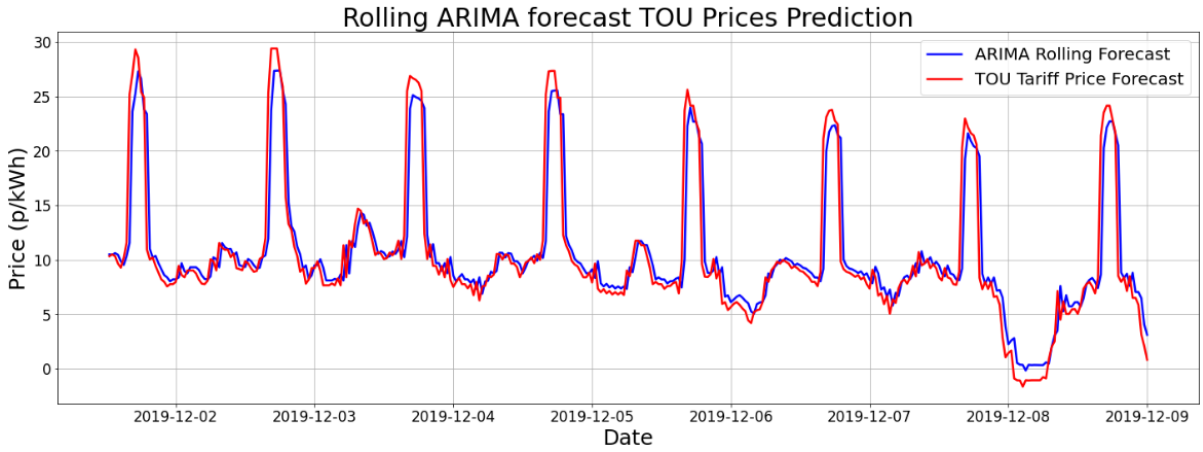

Figure 23 – Rolling ARIMA (Blue) plot for TOU tariff data compared to real tariff data (Red)

4.4.5 Comparison and Discussion

Table 5 provides a quantitative comparison of the models with their respective evaluation metric values. Figure 24 shows how the models compare visually.

As expected, the non-seasonal baseline models produced the least accurate forecast, however surprisingly ARIMA performed worst of all models. This could be due to the data requiring differencing to be performed on it. Even though ADF tests showed that the data was stationary, a seasonal lag may exist that requires further exploration.

Seasonal models proved to perform well if the seasonality of the data remained consistent, and evaluation metric results were quite similar for each model. FBProphet showed lowest MSE and RMSE error, Rolling ARIMA performed best for MAE and SARIMA performed best for MAPE.

Rolling ARIMA visually appears to perform best as it follows the trend closely. As discussed, its MAE is also lowest which agrees with this observation. When running rolling ARIMA, it takes a long time (12-72 hours) depending on the hyperparameter configuration since its window iterates 1 step at a time.

Common issues with the forecasting models are that they don't follow peak trends well, and do not predict outlier sets of data. This is likely due to the training data which didn't have such unexpected values.

Table 5: Summary of error metrics for evaluated forecasting models

Model/Metric	Type	MSE	RMSE	MAE	MAPE (%)
Mean	Baseline	40.79	6.38	4.79	41.66
Naïve		39.53	6.28	4.16	40.13
Seasonal Naïve		9.06	3.00	2.12	21.01
ARIMA	Statistical	40.82	6.39	4.80	41.70
SARIMA		7.62	2.76	1.86	**18.27**
Rolling ARIMA		8.04	2.84	**1.44**	25.78
FBProphet	Additive	**5.39**	**2.32**	1.66	20.30

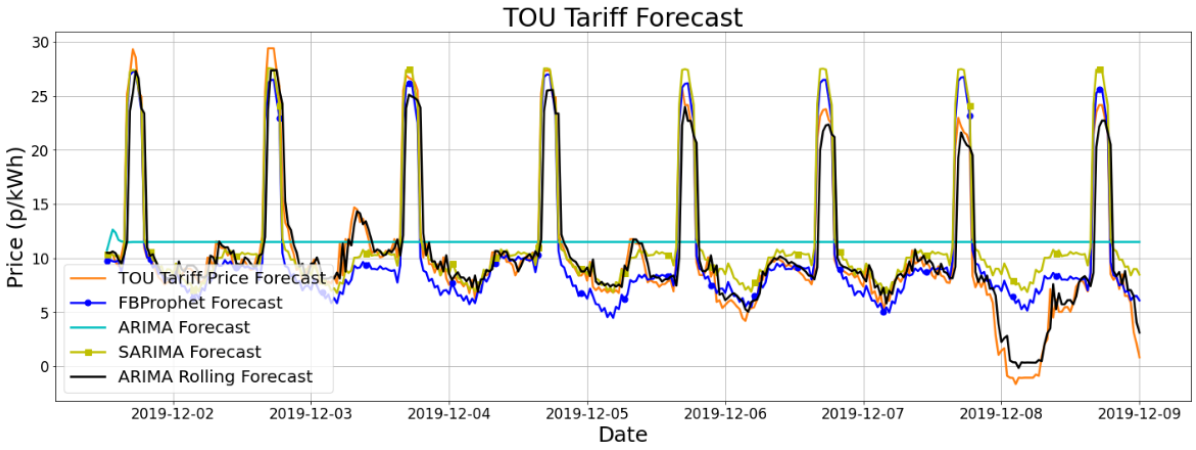

Figure 24 – Compilation of TOU forecast for all complex time series forecasting models used in this project

4.5 Classifier Module

A plot of the classifier module simulating a period of 2 days is shown in Figure 25.

From the plot, it can be seen that the user wakes up at around 7.50am and leaves the house by 9am. They then return home around 4.30-5pm. This result is similar when applied to longer timescale (1 week plot can be found in Appendix 7.17). Although this cannot be confirmed experimentally, empirically, the time periods satisfy the assumptions that the user may be a 9-5 worker.

The periods that the vehicle is believed to be driven is also detected successfully. Since the datasets come from 2 different sources, the users may live different lifestyles, however it is not completely unrealistic since if a household has more than 2 occupants, the lifestyle may also be different.

These results give a good approximation of when users may be at home and when the user is not driving. Using these time periods the scheduling system can then ensure charging only occurs when both the user and the EV is available.

However, current assumptions that the user sleeps from 0-7am may not be valid if the user works the night shift, therefore, user input or customisation may be needed to change this.

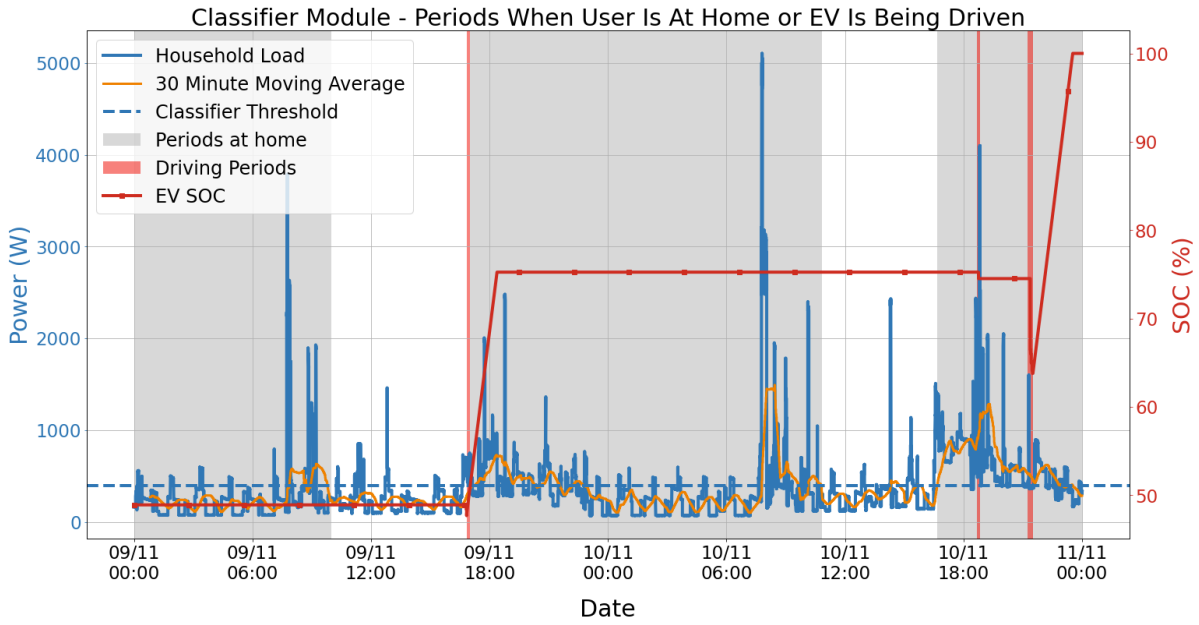

Figure 25 – Classifier module demonstrating work for 2 day period

From the classifier module plot, it can be seen that the algorithm works well to detect regions we believe the user is at home or driving. As there were many assumptions used for both the EV dataset and classifier module implementation, the results can only be approximate however this demonstrates the concept of how the classifier module is meant to function within the smart charging system.

4.6 Scheduling System

A plot of the scheduler module simulating a period of 2 days is shown in Figure 26.

Peak times are defined in legislation to be from 8-11am and 4-10pm on weekdays [98]. So from the plot, it can be seen that smart charging or off-peak periods are highlighted in light green, and the cheapest charging periods recommended by the scheduler are in dark green.

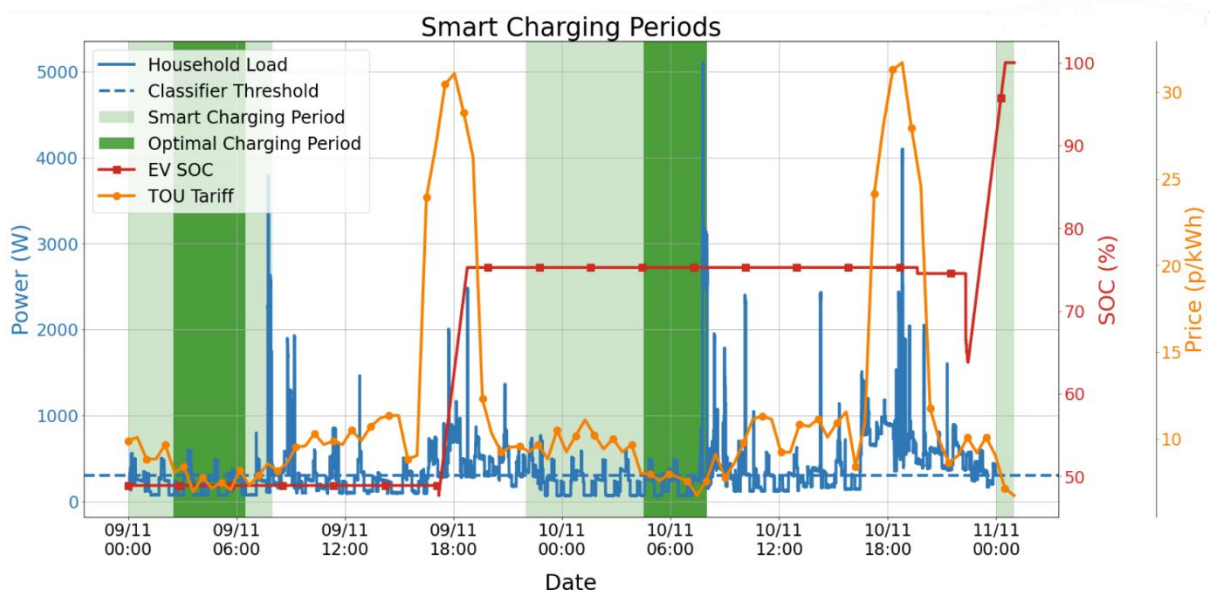

Figure 26 – Scheduling system results showing optimal charging periods for the user for 2 day period. The algorithm shows 2 potential charging periods in these 2 days that with charging periods that

The implemented scheduling system has demonstrated to work albeit with the most basic functionalities of locating cheapest charging periods that can charge the vehicle up to at least the upper limit from the current SOC before the next journey. However, it has been limited by the amount of EV data that was input into the system, making the simulation duration very short so very few scenarios were experienced.

The scheduling system is a rule-based algorithm and was chosen because of its simple and easy to comprehend nature. This is a valid approach as many other literatures also implement this to great effect such as [42][99][100]. Some others have used algorithms such as machine learning or intelligent scatter search (ISS) [33][101] which from quantitative results have performed better.

4.7 Charging Simulation

Now that the user drive cycle, TOU tariff, and household load data has been acquired, even though they all come from different users, we assume that the data collected comes from a user who works a 40 hour per week, day-shift job as explained in Section 3.5 and the vehicle is a 40kWh Nissan Leaf, which was deduced from the EV dataset in Section 3.1.2, so comparisons between the data can be made. To maintain consistency, the day and month of which the data was taken from are all the same but with different year as the data was sourced at different times.

4.7.1 Charging Cost Saved

To evaluate how well the Recommendation Module works, cost to charge for smart vs dumb charging are calculated for a 7-day period. The cost for each optimal charging slot is also compared and averaged with the next upcoming charging slot to find out how much money can be saved. The simulated charging period is shown in Figure 27.

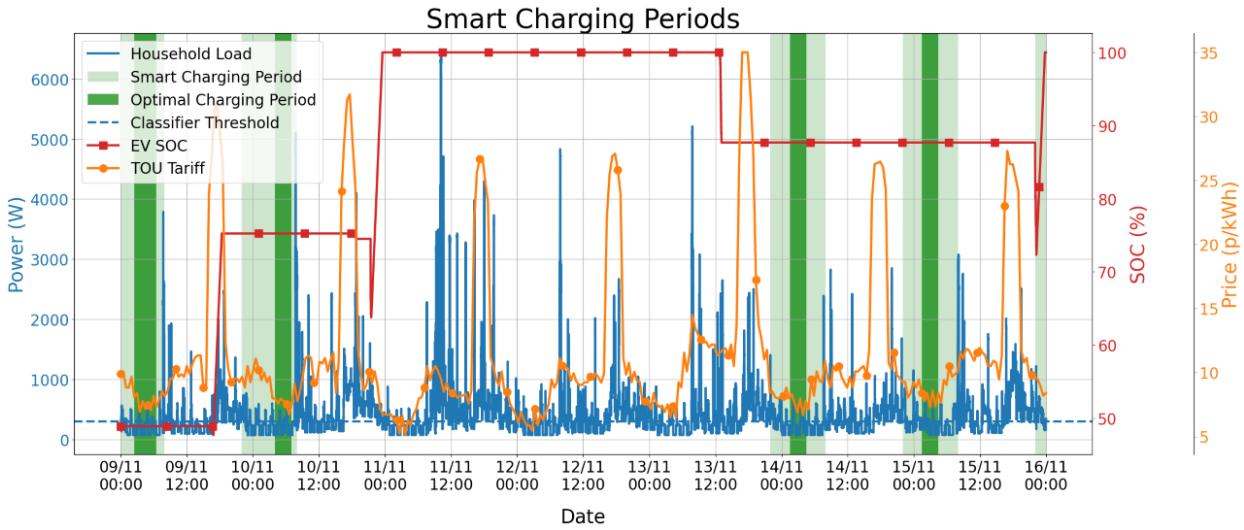

Figure 27 - Scheduling system results showing optimal charging periods for the user for 1 week. Algorithm suggests 4 charging periods (green)

An example of how each calculation is done during the simulation, can be seen in Table 6, which shows the charging that occurs on November 9th. The forecasting module is meant to predict that a journey will occur the next day at around 5pm, followed by a recharge from 48-75% SOC for a charging period of 1 hour 30 mins. Since the charging is done at peak time, the price of energy varies but the average is 28.7 p/kWh. Overall, this would have cost the user £3.10.

The scheduling system however, was able to recommend a 4-hour optimal charging window which would allow the user to charge their vehicle up to the upper limit (80%) with ample time. This period has the cheapest charging cost for the day, so the price of energy is only 7.6 p/kWh and to charge the vehicle, it would only cost them £0.82, reducing the total cost by 74%.

Table 6: Example calculation of potential cost saved using dumb vs smart

'Dumb' Charging	**'Smart' Charging**
Percentage Charged = 75% - 48% = 27% Battery Added = 40kWh * 27% = 10.8kWh	
Average TOU Cost = 28.7p/kWh Overall Cost = $0.287 * 10.8 = £3.10$	Average TOU Cost = 7.6p/kWh Overall Cost = $0.076 * 10.8 = £0.82$

With such a broad recommended charging window, this can encourage the user to use a lower rated charger, for example 3.3kW or 7kW. This will help prolong battery health of the EV [77] because charging batteries produced heat. The faster the charger, the more heat is produced, which can damage battery health [78].

From 11-13th November, the SOC equals 100% and therefore does not allocate any charging periods until necessary. For the charge that the user performs on November 16th, multiple charging windows are recommended that would all be suitable for the user. The average cost of energy over the simulated period is shown in Table 7, and the comparison of cost saved for each day can be found in the Appendix 7.18.

Table 7: Comparison of average cost saved using dumb vs smart charging

	Dumb Charging	**Smart Charging**	**Difference**	**Percentage Difference**
Average Cost of Energy (£)	£1.3	£0.69	£0.61	47%

4.7.2 Analysis and Discussion

The Recommendation Module is simulated for a duration of 7 days. The average cost to charge the vehicle is calculated, the percentage difference (cost saved) yields 47%. This result shows promising reductions in charging cost. However, in such a short simulation period, it is clear that an attempt to generalise the average cost saved has been unsuccessful since a 7-day length of data cannot validate the recommendation algorithm.

The limiting factor in the implementation was the SOC data, which did not record data continuously. Since de-identification methods were used, parts of the journey were not recorded and thus the SOC data was incomplete. More valid results can be formed through increasing the duration of simulation or getting 3 sets of data from the same source.

This result in cost reduction of 47% is comparable with other implemented charging solutions that use rule-based algorithms such as [42][102] which achieved a reduction in cost per kWh of 37.8% and 62.5% respectively.

5 Conclusion and Future Work

This report aims to develop a smart EV charging system based off user drive cycle, TOU and household load data. One of the main objectives this project set out to do, was to obtain real SOC data logged from an EV. The project was partially successful in doing so, using an API that demonstrated it can return SOC of the vehicle as well as other useful data. It also showed strong potential to log data as well. A Recommendation Module was also implemented to function as a smart charging system that suited the user in terms of convenience for charging and minimising charging costs.

Being able to successfully log SOC data is a significant milestone in the field of smart charging systems. As was discussed previously, current research in smart charging systems have only simulated this data. Thus, the results achieved are not entirely representative of a real SOC profile. The logged data from this project can be implemented into other systems to provide better insight into EV charging.

In addition, simulations using this smart charging system yielded a 47% reduction in energy cost in comparison with uncontrolled charging methods. However, as mentioned previously, this value is not fully representative because the datasets used came from different sources. The simulation was also too short to fully validate the outcome, and many assumptions were needed due to the nature of the datasets.

One of the aims of this project was to recommend charging times suiting the convenience of the user. However, it is hard to determine if the user would find these recommended times more convenient. Therefore, it may be necessary to conduct surveys in future research, to establish whether our system practically addresses these user needs.

In conclusion, this project offers an insight into how personalised, smart charging systems can be used to incentivise users to charge more consistently at cheaper times. This has important implications as it can help minimise grid load at peak times, and therefore reduce the investments needed to upgrade current infrastructure to support the future demand of EV charging. In addition, automating charging schedules means that users will no longer need to worry about when or how much they need to charge their vehicle. This will help increase the rate of adoption of EVs to achieve the long-term target of net 0 by 2050.

Future Work

This project has worked on a wide range of tasks and therefore has left a lot of areas to be explored.

Based off the current limitations of this project, future works can work on the following areas:

- Waking the Tesla remotely using the Tesla API, to continue attempts at logging user drive cycle data (SOC, speed, vehicle coordinates)
- Reverse engineer vehicle PIDs and connect the CANedge2 device to cloud servers (see Appendix 7.19, 7.20 for more details)
- Acquire household load data from the same source as the obtained EV SOC data stated above or if this not possible, households that currently own an EV
- Explore different types of scheduling algorithms (machine learning or ISS)
- Extend the charging simulation period to at least 3 months to increase validity of results

6 References

[1] C. R. Creative and A. Stock Photo, “Decarbonising Transport Cover Images,” 2021, Accessed: Apr. 29, 2022. [Online]. Available: www.gov.uk/dft

[2] HM Government, “Net Zero Strategy: Build Back Greener,” Oct. 2021. https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/1033990/net-zero-strategy-beis.pdf (accessed Apr. 29, 2022).

[3] “Transport and environment statistics: Autumn 2021 - GOV.UK.” https://www.gov.uk/government/statistics/transport-and-environment-statistics-autumn-2021/transport-and-environment-statistics-autumn-2021 (accessed Apr. 29, 2022).

[4] “• UK: passenger car GHG emissions 1990-2020 | Statista.” https://www.statista.com/statistics/509066/greenhouse-gas-emissionspassenger-cars-in-the-united-kingdom-uk/ (accessed Apr. 29, 2022).

[5] T. Willis, “The UK’s transition to electric vehicles,” 2020, [Online]. Available: https://www.theccc.org.uk/wp-content/uploads/2020/12/The-UKs-transition-to-electric-vehicles.pdf

[6] C. Lilly, “Electric vehicle market statistics 2022 - How many electric cars in UK ?,” 2022. https://www.nextgreencar.com/electric-cars/statistics/ (accessed Apr. 29, 2022).

[7] D. Powell, “Electric car statistics - data and projections [Updated April 2022] | heycar,” 2022. https://heycar.co.uk/blog/electric-cars-statistics-and-projections (accessed Apr. 29, 2022).

[8] J. A. P. Lopes, F. J. Soares, and P. M. R. Almeida, “Integration of electric vehicles in the electric power system,” *Proceedings of the IEEE*, vol. 99, no. 1, pp. 168–183, Jan. 2011, doi: 10.1109/JPROC.2010.2066250.

[9] M. Alonso, H. Amaris, J. G. Germain, and J. M. Galan, “Optimal Charging Scheduling of Electric Vehicles in Smart Grids by Heuristic Algorithms,” *Energies 2014, Vol. 7, Pages 2449-2475*, vol. 7, no. 4, pp. 2449–2475, Apr. 2014, doi: 10.3390/EN7042449.

[10] F. G. Dias, D. Scoffield, M. Mohanpurkar, R. Hovsapian, and A. Medam, “Impact of controlled and uncontrolled charging of electrical vehicles on a residential distribution grid,” *2018 International Conference on Probabilistic Methods Applied to Power Systems, PMAPS 2018 - Proceedings*, Aug. 2018, doi: 10.1109/PMAPS.2018.8440511.

[11] L. Pickett, J. Winnett, D. Carver, and P. Bolton, “Electric vehicles and infrastructure,” 2021.

[12] H. Government, “Electric Vehicle Smart Charging,” 2021, Accessed: Apr. 29, 2022. [Online]. Available: www.nationalarchives.gov.uk/doc/opengovernment-licence/version/3/

[13] “Agile Octopus | Octopus Energy.” https://octopus.energy/agile/ (accessed Apr. 29, 2022).

[14] “ev.energy | ev.energy customers reduce peak demand by 80% as part of Project Shift - ev.energy.” https://ev.energy/news/ev-energy-customers-help-uk-grid-shift-80-peak-energy-demand-during-project-shift-initiative/ (accessed Apr. 29, 2022).

[15] S. Hutson, “Peak and off-peak electricity times,” May 2021. https://www.comparethemarket.com/energy/content/is-energy-cheaper-at-night/ (accessed Apr. 29, 2022).

[16] C. Argue, “EV Battery Health - What 6,000 Batteries Tell Us | Geotab,” Dec. 2019. https://www.geotab.com/uk/blog/ev-battery-health/ (accessed Apr. 29, 2022).

[17] K. Qian, C. Zhou, M. Allan, and Y. Yuan, “Modeling of load demand due to EV battery charging in distribution systems,” *IEEE Transactions on Power Systems*, vol. 26, no. 2, pp. 802–810, May 2011, doi: 10.1109/TPWRS.2010.2057456.

[18] M. Müller, F. Samweber, and P. Leidl, “Impact of different charging strategies for electric vehicles on their grid integration,” pp. 41–55, 2017, doi: 10.1007/978-3-658-19293-8_5.

[19] A. J. Cheng, B. Tarroja, B. Shaffer, and S. Samuelsen, “Comparing the emissions benefits of centralized vs. decentralized electric vehicle smart charging approaches: A case study of the year 2030 California electric grid,” *Journal of Power Sources*, vol. 401, pp. 175–185, Oct. 2018, doi: 10.1016/J.JPOWSOUR.2018.08.092.

[20] T. Simolin, K. Rauma, P. Järventausta, and A. Rautiainen, “Optimised controlled charging of electric vehicles under peak power-based electricity pricing,” *IET Smart Grid*, vol. 3, no. 6, pp. 751–759, Dec. 2020, doi: 10.1049/IET-STG.2020.0100.

[21] A. R. Abul, fotouh El, W. AFatah Mohamed, W. AFatah Mohamed Uncoordinated vs Coor-, and F. El, “dinated Charging of Electric Vehicles in Distribution Systems Performance,” *International Journal of Engineering and Information Systems*, vol. 2017, no. 6, pp. 54–65, 2017, Accessed: Apr. 29, 2022. [Online]. Available: www.ijeais.org54

[22] M. Liu, P. McNamara, R. Shorten, and S. McLoone, “Residential electrical vehicle charging strategies: the good, the bad and the ugly,” *Journal of Modern Power Systems and Clean Energy*, vol. 3, no. 2, pp. 190–202, Jan. 2015, doi: 10.1007/S40565-015-0122-2/TABLES/3.

[23] Q. Wang, X. Liu, J. Du, and F. Kong, “Smart Charging for Electric Vehicles: A Survey From the Algorithmic Perspective”.

[24] P. Richardson, D. Flynn, and A. Keane, “Local versus centralized charging strategies for electric vehicles in low voltage distribution systems,” *IEEE Transactions on Smart Grid*, vol. 3, no. 2, pp. 1020–1028, 2012, doi: 10.1109/TSG.2012.2185523.

[25] A. Amin *et al.*, “A Review of Optimal Charging Strategy for Electric Vehicles under Dynamic Pricing Schemes in the Distribution Charging Network,” *Sustainability 2020, Vol. 12, Page 10160*, vol. 12, no. 23, p. 10160, Dec. 2020, doi: 10.3390/SU122310160.

[26] L. Zhang, V. Kekatos, and G. B. Giannakis, “Scalable Electric Vehicle Charging Protocols,” *IEEE Transactions on Power Systems*, vol. 32, no. 2, pp. 1451–1462, Mar. 2017, doi: 10.1109/TPWRS.2016.2582903.

[27] I. K. A. Aswantara, K. S. Ko, and D. K. Sung, “A centralized EV charging scheme based on user satisfaction fairness and cost,” *2013 IEEE Innovative Smart Grid Technologies - Asia, ISGT Asia 2013*, 2013, doi: 10.1109/ISGT-ASIA.2013.6698730.

[28] Z. J. Lee *et al.*, “Adaptive Charging Networks: A Framework for Smart Electric Vehicle Charging,” *IEEE Transactions on Smart Grid*, vol. 12, no. 5, pp. 4339–4350, Dec. 2020, doi: 10.48550/arxiv.2012.02636.

[29] S. S. Ravi and M. Aziz, “Utilization of Electric Vehicles for Vehicle-to-Grid Services: Progress and Perspectives,” *Energies 2022, Vol. 15, Page 589*, vol. 15, no. 2, p. 589, Jan. 2022, doi: 10.3390/EN15020589.

[30] M. Liu, P. K. Phanivong, Y. Shi, and D. S. Callaway, “Decentralized Charging Control of Electric Vehicles in Residential Distribution Networks,” 2017.

[31] S. Xu, Z. Yan, D. Feng, and X. Zhao, “Decentralized charging control strategy of the electric vehicle aggregator based on augmented Lagrangian method,” *International Journal of Electrical Power & Energy Systems*, vol. 104, pp. 673–679, Jan. 2019, doi: 10.1016/J.IJEPES.2018.07.024.

[32] M. G. Vaya and G. Andersson, “Plug-in Electric Vehicle charging approaches: Centralized versus decentralized and strategic versus cooperative,” *2015 IEEE Eindhoven PowerTech, PowerTech 2015*, Aug. 2015, doi: 10.1109/PTC.2015.7232260.

[33] T. Mao, X. Zhang, and B. Zhou, “Intelligent Energy Management Algorithms for EV-charging Scheduling with Consideration of Multiple EV Charging Modes,” *Energies 2019, Vol. 12, Page 265*, vol. 12, no. 2, p. 265, Jan. 2019, doi: 10.3390/EN12020265.

[34] N. Chen, C. W. Tan, and T. Q. S. Quek, “Electric vehicle charging in smart grid: Optimality and valley-filling algorithms,” *IEEE Journal on Selected Topics in Signal Processing*, vol. 8, no. 6, pp. 1073–1083, Dec. 2014, doi: 10.1109/JSTSP.2014.2334275.

[35] “Pioneering electric vehicle study shows up to 95% of consumers happy to use smart charging.” https://es.catapult.org.uk/news/pioneering-electric-vehicle-study-shows-up-to-95-of-consumers-happy-to-use-smart-charging/ (accessed Apr. 29, 2022).

[36] S. Hardman *et al.*, “A review of consumer preferences of and interactions with electric vehicle charging infrastructure,” *Transportation Research Part D: Transport and Environment*, vol. 62, pp. 508–523, Jul. 2018, doi: 10.1016/J.TRD.2018.04.002.

[37] Y.-Y. Wang *et al.*, “Consumer Preferences for Electric Vehicle Charging Infrastructure Based on the Text Mining Method,” *Energies 2021, Vol. 14, Page 4598*, vol. 14, no. 15, p. 4598, Jul. 2021, doi: 10.3390/EN14154598.

[38] B. Wang, B. Hu, C. Qiu, P. Chu, and R. Gadh, “EV charging algorithm implementation with user price preference,” *2015 IEEE Power and Energy Society*

Innovative Smart Grid Technologies Conference, ISGT 2015, Jun. 2015, doi: 10.1109/ISGT.2015.7131895.

[39] "Agile pricing explained | Octopus Energy." https://octopus.energy/blog/agile-pricing-explained/ (accessed Apr. 29, 2022).

[40] Roseanne, "No Driveway? You Can Still Have an Electric Car | Pod Point," 2021. https://pod-point.com/electric-car-news/electric-car-no-driveway (accessed Apr. 29, 2022).

[41] pwc, "Charging ahead! The need to upscale UK electric vehicle charging infrastructure," 2018.

[42] K.Shyang, H.Koo, and R.Chahine, "Smart Residential EV Charging System Based On Dynamic Time-Of-Use (TOU) Tariff," 2021.

[43] V.Barreto, "What is OBDII? History of on-board diagnostics ," 2020. https://www.geotab.com/uk/blog/obd-ii/ (accessed Apr. 29, 2022).

[44] autopi, "CAN Bus Protocol: The Ultimate Guide (2022)," 2021. https://www.autopi.io/blog/can-bus-explained/ (accessed Apr. 29, 2022).

[45] CSS Electronics, "CAN Bus Explained - A Simple Intro [2022 | ." https://www.csselectronics.com/pages/can-bus-simple-intro-tutorial (accessed Apr. 29, 2022).

[46] OBD Experts, "OBD II Protocols Explained ." https://www.obdexperts.co.uk/obd-ii-protocols-explained/ (accessed Apr. 29, 2022).

[47] CSS Electronics, "OBD2 Explained - A Simple Intro [2022 |." https://www.csselectronics.com/pages/obd2-explained-simple-intro (accessed Apr. 29, 2022).

[48] AutoPi, "What is OBD2 and How Does it Work? ," 2020. https://www.autopi.io/blog/what-is-obd-2/ (accessed Apr. 29, 2022).

[49] "DBC Introduction — Open Vehicles documentation." https://docs.openvehicles.com/en/latest/components/vehicle_dbc/docs/dbc-primer.html (accessed Apr. 29, 2022).

[50] CSS Electronics, "CAN DBC File Explained ." https://www.csselectronics.com/pages/can-dbc-file-database-intro (accessed Apr. 29, 2022).

[51] Tesla UK, "Artificial Intelligence & Autopilot ." https://www.tesla.com/en_GB/AI (accessed Apr. 29, 2022).

[52] Mobility Insider, "What Is an API in Automotive?," 2021. https://www.aptiv.com/en/insights/article/what-is-an-api-in-automotive (accessed Apr. 29, 2022).

[53] IBM Cloud, "Identifying API use cases: Automotive industry".

[54] Red Hat, “What is API security?,” 2019. https://www.redhat.com/en/topics/security/api-security#why-is-api-security-important (accessed Apr. 29, 2022).

[55] Experian, “Fixed & Variable Energy Explained.” https://www.experian.co.uk/consumer/energy/guides/fixed-variable.html (accessed Apr. 29, 2022).

[56] S.Hutson, “How Much Can I Save With Economy 7 & 10 Tariffs?,” 2020. https://www.comparethemarket.com/energy/content/how-much-can-i-save-with-economy-7-and-10-tariffs/ (accessed Apr. 29, 2022).

[57] Energy Saving Trust, “Time of use tariffs: all you need to know.” https://energysavingtrust.org.uk/time-use-tariffs-all-you-need-know/ (accessed Apr. 29, 2022).

[58] Solar Choice Staff, “How do I use electricity during the day - the load curve,” 2010. https://www.solarchoice.net.au/blog/how-do-i-use-electricity-throughout-the-day-the-load-curve/ (accessed Apr. 29, 2022).

[59] S. Barker, A. Mishra, D. Irwin, P. Shenoy, and J. Albrecht, “SmartCap: Flattening peak electricity demand in smart homes,” *2012 IEEE International Conference on Pervasive Computing and Communications, PerCom 2012*, pp. 67–75, 2012, doi: 10.1109/PERCOM.2012.6199851.

[60] Smart Energy GB, “Time-of-use tariffs: the benefits.” https://www.smartenergygb.org/smart-meter-benefits/benefits-for-you/time-of-use-tariffs-the-benefits (accessed Apr. 29, 2022).

[61] S.Hutson, “Energy tariffs explained ,” 2021. https://www.comparethemarket.com/energy/content/energy-tariffs-explained/ (accessed Apr. 29, 2022).

[62] “Agile pricing explained | Octopus Energy.” https://octopus.energy/blog/agile-pricing-explained/ (accessed Apr. 29, 2022).

[63] M.Zhao, “Stationarity and Forecastability ,” 2019. https://ming-zhao.github.io/Business-Analytics/html/docs/time_series/stationarity.html (accessed Apr. 29, 2022).

[64] M.Gupta, “Why Time Series has to be STATIONARY ??? | by Mehul Gupta | Data Science in your pocket | Medium,” 2019. https://medium.com/data-science-in-your-pocket/why-time-series-has-to-be-stationary-37ca8800ddf (accessed Apr. 29, 2022).

[65] M.Heath, “Should You Use Rolling Forecasts? Weighing the Pros & Cons.” https://www.us-analytics.com/hyperionblog/should-you-use-rolling-forecasts-weighing-the-pros-and-cons (accessed Apr. 29, 2022).

[66] CSS Electronics, “CANedge2: 2x CAN Bus Data Logger (SD + WiFi) .” https://www.csselectronics.com/products/can-bus-data-logger-wifi-canedge2 (accessed Apr. 29, 2022).

[67] CSS Electronics, “OBD2 Data Logger - Easily Record & Visualize Your Car Data .” https://www.csselectronics.com/pages/obd2-data-logger-sd-memory-convert (accessed Apr. 29, 2022).

[68] G. S. Oh, D. J. Leblanc, and H. Peng, “Vehicle Energy Dataset (VED), A Large-scale Dataset for Vehicle Energy Consumption Research,” 2019, Accessed: Apr. 29, 2022. [Online]. Available: https://github.com/gsoh/VED.

[69] G. Oh, D. J. Leblanc, and H. Peng, “Vehicle Energy Dataset (VED), A Large-Scale Dataset for Vehicle Energy Consumption Research,” *IEEE Transactions on Intelligent Transportation Systems*, Apr. 2020, doi: 10.1109/TITS.2020.3035596.

[70] Statista, “• Electric vehicles: market share by type EU28 2010-2017 ,” 2019. https://www.statista.com/statistics/1010373/electric-vehicles-market-share-by-type-eu28/ (accessed Apr. 29, 2022).

[71] RAC Drive, “Electric car charging – how it works and how much it costs .” https://www.rac.co.uk/drive/electric-cars/charging/electric-car-charging-how-it-works-and-how-much-it-costs/ (accessed Apr. 29, 2022).

[72] Pod Point, “Charging an Electric Car at Home,” 2022. https://pod-point.com/guides/driver/charging-electric-car-at-home (accessed Apr. 29, 2022).

[73] “How to charge your electric car at home | Autocar,” Dec. 2021. https://www.autocar.co.uk/car-news/advice-electric-cars/how-charge-your-electric-car-home (accessed Apr. 29, 2022).

[74] “Nissan LEAF (2018) Charging Guide | Pod Point,” 2022. https://pod-point.com/guides/vehicles/nissan/2018/leaf (accessed Apr. 29, 2022).

[75] S.Edelstein, “2017 electric cars with more than 100 miles of range ,” 2017. https://www.greencarreports.com/news/1107455_2017-electric-cars-with-more-than-100-miles-of-range (accessed Apr. 29, 2022).

[76] M. Skowron, “Smart EV Charging Habits,” May 2020. https://blog.greenenergyconsumers.org/blog/smart-ev-charging-habits (accessed Apr. 29, 2022).

[77] M. Kane, “33 kWh BMW i3 And 30 kWh Nissan LEAF Fast Charging Comparison,” 2016. https://insideevs.com/news/330285/33-kwh-bmw-i3-and-30-kwh-nissan-leaf-fast-charging-comparison/ (accessed Apr. 29, 2022).

[78] “BU-808: How to Prolong Lithium-based Batteries - Battery University,” 2021. https://batteryuniversity.com/article/bu-808-how-to-prolong-lithium-based-batteries (accessed Apr. 29, 2022).

[79] C.Lilly, “Nissan Leaf 24 kWh & 30 kWh Charging Guide,” 2022. https://www.zap-map.com/charge-points/nissan-leaf-2430kwh-charging-guide/ (accessed Apr. 29, 2022).

[80] “Tesla API - Tesla API.” https://www.teslaapi.io/ (accessed Apr. 29, 2022).

[81] “Tesla Owner JSON API · Apiary.” https://akrion.docs.apiary.io/#reference/authentication/tokens/get-an-access-token?console=1 (accessed Apr. 29, 2022).

[82] “Introduction - Tesla JSON API (Unofficial).” https://tesla-api.timdorr.com/ (accessed Apr. 29, 2022).

[83] “CoolTechYT/Tesla-Dashboard: A dashboard built with React, Node.js, and Express.js to visualize some data from Tesla’s Private API.” https://github.com/CoolTechYT/Tesla-Dashboard (accessed Apr. 29, 2022).

[84] David Murray, Lina Stankovic, and Vladimir Stankovic, “REFIT: Electrical Load Measurements (Cleaned),” *Uniwersytet śląski*, 2013. https://sbc.org.pl/dlibra/publication/99008/edition/93276/synteza-i-aktywnosc-biologiczna-nowych-analogow-tiosemikarbazonowych-chelatorow-zelaza-serda-maciej?language=en (accessed Apr. 29, 2022).

[85] D. Murray, L. Stankovic, and V. Stankovic, “Data Descriptor: An electrical load measurements dataset of United Kingdom households from a two-year longitudinal study Background & Summary”, doi: 10.1038/sdata.2016.122.

[86] “Download Historical Pricing Data - Energy Stats UK.” https://www.energy-stats.uk/download-historical-pricing-data/ (accessed Apr. 29, 2022).

[87] H.Chen, “Henchen99/BEng-Project: Repository for BEng project on a Smart Electric Vehicle Charging System Based on Time-Of-Use Tariff And User Drive Cycle Data,” 2022. https://github.com/Henchen99/BEng-Project (accessed Apr. 29, 2022).

[88] “• Average working hours UK 2022 | Statista.” https://www.statista.com/statistics/280763/average-working-hours-uk/ (accessed Apr. 30, 2022).

[89] H. R. Lieberman, S. Agarwal, J. A. Caldwell, and V. L. Fulgoni, “Demographics, sleep, and daily patterns of caffeine intake of shift workers in a nationally representative sample of the US adult population,” *Sleep*, vol. 43, no. 3, Mar. 2020, doi: 10.1093/SLEEP/ZSZ240.

[90] E.Suni and K.Truong, “Sleep Statistics - Facts and Data About Sleep 2022 | Sleep Foundation,” 2022. https://www.sleepfoundation.org/how-sleep-works/sleep-facts-statistics (accessed Apr. 30, 2022).

[91] “Decoding the i3’s PIDs - Page 3 - i3 - A Better Route Planner,” 2019. https://forum.abetterrouteplanner.com/topic/317-decoding-the-i3s-pids/page/3/ (accessed Apr. 29, 2022).

[92] “Car Alarm Goes Off after Locking - BMW i3 Forum,” 2021. https://www.mybmwi3.com/forum/viewtopic.php?t=17525 (accessed Apr. 29, 2022).

[93] L.Najman, “Testing the Vampire Drain Problem & Other Tesla Folklore,” May 2021. https://www.recurrentauto.com/research/tesla-vampire-drain (accessed Apr. 29, 2022).

[94] "Tesla binding keeps car from sleeping ie. Vampire Drain · Issue #2290 · openhab/openhab-addons," 2017. https://github.com/openhab/openhab-addons/issues/2290 (accessed Apr. 29, 2022).

[95] C. Argue, "To what degree does temperature impact EV range? | Geotab," May 2020. https://www.geotab.com/uk/blog/ev-range/ (accessed Apr. 29, 2022).

[96] "Wholesale energy prices and Agile Octopus | Octopus Energy." https://octopus.energy/blog/wholesale-price-watch/ (accessed Apr. 29, 2022).

[97] J. Brownlee, "How to Make Baseline Predictions for Time Series Forecasting with Python," 2016. https://machinelearningmastery.com/persistence-time-series-forecasting-with-python/ (accessed Apr. 29, 2022).

[98] H. Government, "Electric Vehicle Smart Charging," 2021, Accessed: Apr. 30, 2022. [Online]. Available: www.nationalarchives.gov.uk/doc/opengovernment-licence/version/3/

[99] J. Lundblad and R. S. Duvernoy, "Development of a Smart Charging Algorithm for Electric Vehicles at Home," 1900, Accessed: Apr. 30, 2022. [Online]. Available: http://www.teknat.uu.se/student

[100] B. v. Mbuwir, L. Vanmunster, K. Thoelen, and G. Deconinck, "A hybrid policy gradient and rule-based control framework for electric vehicle charging," *Energy and AI*, vol. 4, p. 100059, Jun. 2021, doi: 10.1016/J.EGYAI.2021.100059.

[101] Z. Yi and D. Scoffield, "A Data-Driven Framework for Residential Electric Vehicle Charging Load Profile Generation," *2018 IEEE Transportation and Electrification Conference and Expo, ITEC 2018*, pp. 220–225, Aug. 2018, doi: 10.1109/ITEC.2018.8450228.

[102] B. v. Mbuwir, L. Vanmunster, K. Thoelen, and G. Deconinck, "A hybrid policy gradient and rule-based control framework for electric vehicle charging," *Energy and AI*, vol. 4, p. 100059, Jun. 2021, doi: 10.1016/J.EGYAI.2021.100059.

[103] "3.1 Some simple forecasting methods | Forecasting: Principles and Practice (2nd ed)." https://otexts.com/fpp2/simple-methods.html (accessed May 04, 2022).

[104] "The Math of Prophet. Breaking down the Equation behind… | by Winston Robson | Future Vision | Medium." https://medium.com/future-vision/the-math-of-prophet-46864fa9c55a (accessed May 04, 2022).

[105] CSS Electronics, "Documentation - CANedge, CLX000 and CANmod." https://www.csselectronics.com/pages/can-bus-hardware-software-docs (accessed Apr. 29, 2022).

[106] "Wireshark · Go Deep." https://www.wireshark.org/ (accessed Apr. 29, 2022).

7 Appendix

7.1 CAN ID message structure

Figure 1 - Standard CAN frame format source [45]

7.2 OBD2

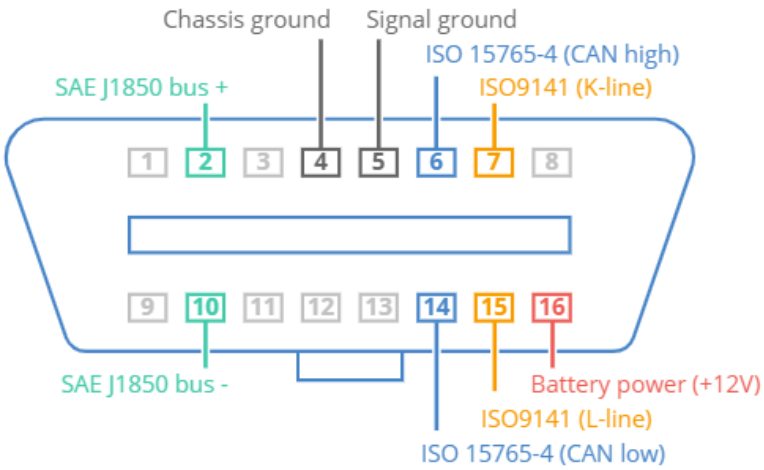

Figure 2 - OBD2 16 pin connector typically found underneath the steering wheel [47]

7.3 DBC

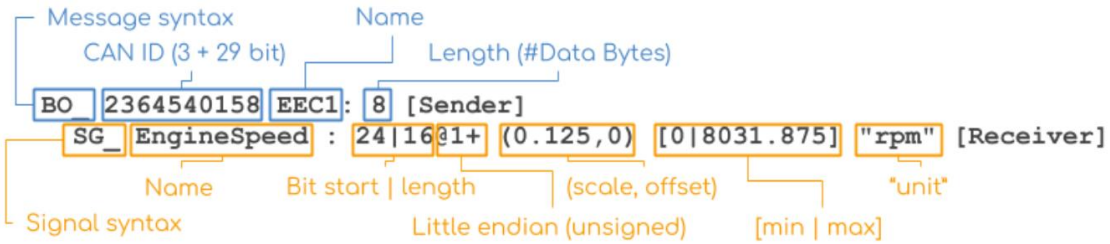

Figure 3 - DBC file structure used to decode raw CAN data into physical values [50]

7.4 Evaluation Metrics

Scale Dependent

- Mean Squared Error (MSE)

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

Where:

1. $\boldsymbol{n}$ is the total number of values
2. $\boldsymbol{y_i}$ is the actual value observation $\boldsymbol{i}$
3. $\boldsymbol{\hat{y}_i}$ is the predicted value at observation $\boldsymbol{i}$

Characteristics:

- Puts emphasis on outliers and large errors

- Squaring means units are lost

- Root Mean Squared Error (RMSE)

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

Characteristics:

- Similar to MSE, but avoids the loss of unit

- Sensitive to outliers

- Mean Absolute Error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|^2$$

Characteristics:

- Calculates the residual for each data point (Difference between predicted and observed)

- Good for assessing a single type of forecast (same units)

- Isn't affected by outliers

Percentage Error

- Mean Absolute Percentage Error (MAPE)

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \widehat{y_i}}{y_i} * 100\right|$$

Characteristics:

- Scale-independent

- Easily interpretable, percentage equivalent of MAE

- Is affected more by negative errors than positive leading to asymmetry

7.5 PIDs

Mode 1	Get current data (RPM, Speed, Fuel Level, Engine Load, etc)
Mode 2	Get data that existed at time of last DTC
Mode 3	Get Diagnostic Trouble Codes (DTCs)
Mode 4	Clear Diagnostic Trouble Codes (DTCs)
Mode 5	Oxygen sensor monitoring test
Mode 6	On-board monitoring, other tests
Mode 7	Pending Diagnostic Trouble Codes
Mode 8	Control operation of on-board system
Mode 9	Request Vehicle information (eg VIN)
Mode 10	Permanent Diagnostic Trouble Codes

Figure 4 - List of OBD2 modes that is used to receive and request information about the vehicle [47]

7.6 CANedge2 Procedure

Originally, EV data was to be extracted from a BMW i3 using a CSS electronics' CANedge2 CAN bus data logger from [36]. There are 4 main steps in the procedure to log data using the CANedge2 can be found below [37]:

Configuration

1. Using 'OBD2 configuration file' and 'OBD2 PID transmit list' provided by CSS electronics [38] as a template, the "ID", "data" and "name" variables are changed according to what data is required
2. The transmit list along with Schema and UISchema files also provided by CSS electronics [39] are then uploaded onto config editor [40]

3. In the config editor make sure to keep mode as ‘normal’ and then click “Review changes” to download the config file onto the SD card before inserting it into the CANedge2.

Record

1. Turn on the vehicle
2. Plug the OBD2 cable adapter into the slot under the left side of the steering wheel
3. Connect a DB9 connector into CH1 of the CANedge device and the following LEDs should show (figure 3):

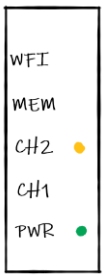

Figure 5 - LED configuration on CANedge2 when monitoring data

4. Wait a few minutes for data to be logged

Transfer

1. Disconnect the CANedge from the vehicle
2. Extract the SD card and insert it back into the PC
3. Transfer the logged data which should be in a LOG/ folder onto your PC. A new folder is created each time the device begins to log data. Each file in the folder contains the data that has just been recorded in a .MF4 file format with a max size of 51MB. Therefore the longer the recording session, the more files will be created

Process

1. The data can be visualised using the provided Asammdf GUI [41]
2. Open a .MF4 file on Asammdf
3. In the ‘Bus Logging’ tab, load in a DBC file provided by CSS Electronics [41] to finally extract the signal (Figure 4)

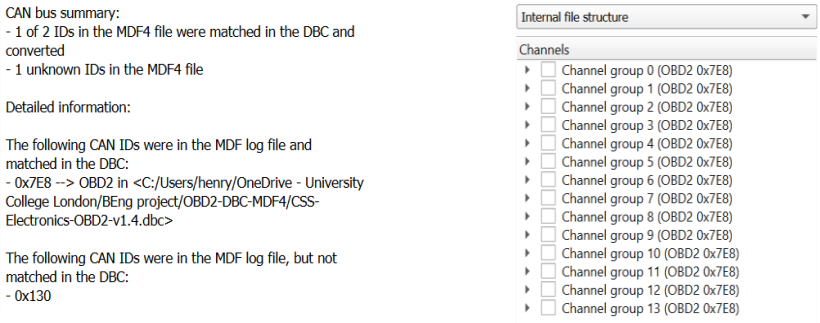

Figure 6 - Output shown in Asammdf if data logged and decoded is successful

7.7 Vehicle Energy Dataset (VED)

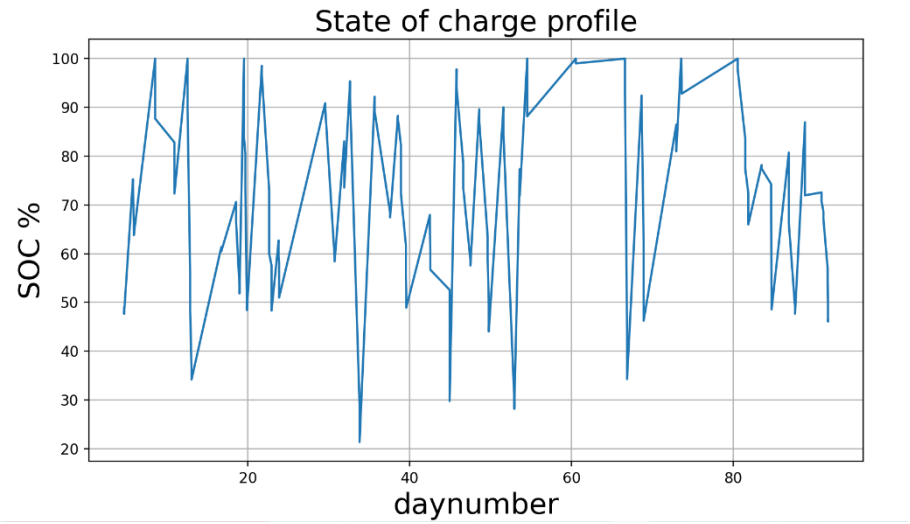

Figure 7 - Plot of SOC vs number of days passed since 01/17/2017 without timescale formatting, which results in the previous value trying to "catch up" with the new value hence the zigzag lines

7.8 Travel routes

273	5.889617213	0	-23.5	66.46341705	380	42.27361556	-83.77983917
978	5.889617213	603600	40	63.78048706	385	42.2542625	-83.72893083

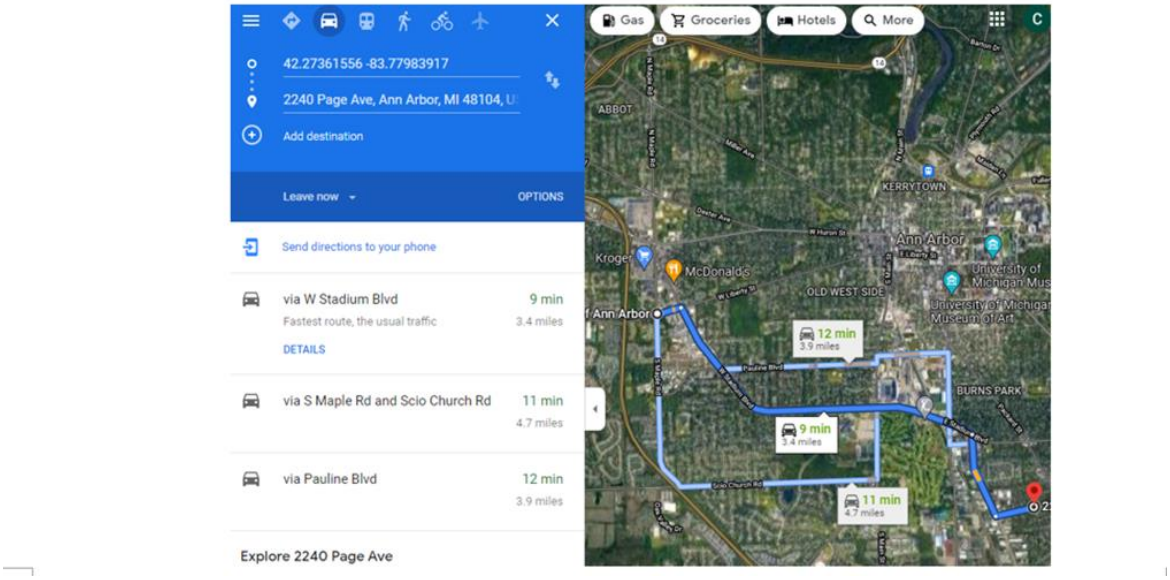

Figure 8 - Journey made on day 5 visualised using Google maps, the driver took the middle route

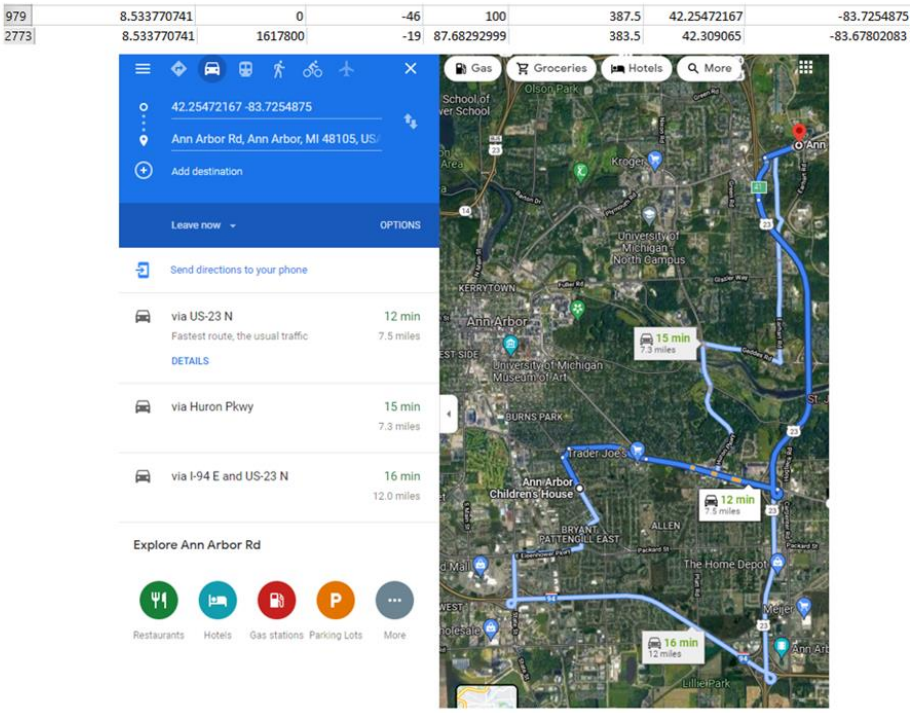

Figure 9 - Journey made on day 8 visualised using Google maps, the driver took the dark blue route

Table 1: A summary of all the trips taken during the first month of recording

Journey Day	Start location	End location	Start SOC %	END SOC %	Distance Travelled miles	Est time (min)	Actual time (min)
4.7	East Eisenhower Pkwy 42.2439327778 -83.7438258333	East Eisenhower Pkwy 42.2446325 -83.72813611	48.9024429321	47.9268302917	1.3	4	1.98
5.8	3183 Birch Hollow Dr 42.2389716667 -83.7201641667	1381 King George Blvd 42.2455263889 -83.7250397222	75.2439117432	74.5121917725	0.7	2	1.25
5.9	3638 W Liberty Rd 42.2736155556 -83.7798391667	2240 Page Ave, 42.2542625 -83.7289308333	66.4634170532	63.7804870605	3.4	11	10.06
8.5	2309 Packard St, , MI 48104, 42.2547216667 -83.7254875	Uom East Medical Campus 42.309065 -83.6780208333	100	87.6829299927	7.5	14	26.96 (route taken was not same as maps)
10.9	Earhart rd (junction) 42.29718 -83.6817902778	2511 Packard St 42.2491316667 -83.7209908333	82.8048782349	72.3170776367	6.2	14	16.76
12.5	Esch Ave/ King George Blvd 1420 42.2464991667 -83.7248202778	Uom East Medical Campus 42.3071336111 -83.6770194444	100	89.5121994019	7.9	15	11.9

12.8	2000-2034 Chaucer Dr 42.2565786111 -83.7577166667	1387 Page Ave 42.2523519444 -83.7282283333	56.7073173523	48.2926864624	2.6	8	30.96 (driver seems to hang around a high school for 23.5 mins before taking the journey shown on google maps)
12.9	Pauline Blvd (junction) 42.2680036111 -83.7710755556	Lake trust credit union 42.2739155556 -83.7763566667	41.0975646973	40.3658561707	0.5	2	1
13.0	999-801 W Stadium Blvd (junction) 42.2639958333 -83.7578430556	2323 Page Ave, Bouncy daycare In Home 42.2532580556 -83.7283138889	36.4634170532	34.1463432312	2.0	6	5.2
16.7	1501 E Stadium Blvd, , MI 48104, 42.2610286111 -83.7299508333	581-495 S Division St, , MI 48104, 42.2758036111 -83.7442119444	61.3414649963	60.4878082275	1.3	5	3.9
18.5	1779 Packard St Chase bank/mortgage 42.2545933333 -83.7258502778	West William St. (Junction) 42.2779886111 -83.7496102778	70.6097564697	66.4634170532	2.1	8	7.7
18.9	Packard St (middle of road) 42.2451419444 -83.7028136111	King George Blvd 1410 42.2463358333 -83.7247355556	53.2926864624	51.8292694092	1.2	3	2.2
19.7	UoM East medical Campus 42.30790639 -83.683175	Broadway St (highway) 42.28729861 -83.74283361	80.48780823	73.29268646	3.6	10	10.6
19.8	East Ellsworth Rd (junction) 42.22993694 -83.71332694	1420 King George Blvd 42.24654111 -83.72458389	57.92683029	56.09756088	1.7	4	6.15
19.9	1405-1401 S Main St 42.26406028 -83.75005694	2499-2401 Page Ave 42.2525925 -83.72823111	50.12195587	48.4146347	1.7	5	4.3
21.7	1849-1821 W Stadium Blvd (Junction) 42.26423583 -83.75039	West Stadium Blvd (Junction) 42.26770944 -83.77092472	98.53659058	96.58537292	1.2	3	1.9
22.6	Ponds Dr (intersection between Packard and King George) 42.2495275 -83.72106861	UoM East medical campus 42.30721167 -83.67707083	72.92683411	60.00000381	7.7	15	12

22.9	UoM East medical campus 42.30927361 -83.677395	2323 Page Ave Bouncy Daycare in Home 42.253435 -83.72843417	57.56097794	48.29268646	7.6	21	26.5
23.8	East Eisenhower Pkwy 42.24445833 -83.72910333	Ann Arbor-Saline Rd (Highway) 42.24436472 -83.7648725	62.68292999	59.63415146	2.0	6	4
23.9	Ann Arbor-Saline Rd (Highway) (cont) 42.24712583 -83.76162444	124 East Washington Street 42.28031833 -83.74733417	58.04878616	50.9756088257	2.6	9	40 (driver takes detour passed Pioneer High school from 4.9min – 30.8min(car idles)) Coordinates fluctuate when car idles maybe because satellite not that accurate?
29.5	Huron Pkway Speedway Petrol station (Junction/traffic lights) 42.30282167 -83.70447194	Plymouth green crossings shopping centre 42.30629361 -83.69391722	90.85366058	89.8780593872	0.8	3	2.7
30.7	1405-1401 S Main St (Junction) 42.26419083 -83.75025694	299 East William Street 42.27790194 -83.74643528	59.14634323	58.41463852	1.1	5	4

7.9 SOC Range Estimate

Table 2- SOC and range values summarised to approximate range of the vehicle

SOC Start %	SOC End %	Difference %	Miles Driven	Approx Range (miles)
48.9024429321	47.9268302917	0.975613	1.3	133.2496
75.2439117432	74.5121917725	0.73172	0.7	95.6650
66.4634170532	63.7804870605	2.68293	3.4	126.7271
100	87.6829299927	12.31707	7.5	60.8911
82.8048782349	72.3170776367	10.4878	6.2	59.1163
100	89.5121994019	10.4878	7.9	75.3256
56.7073173523	48.2926864624	8.414631	2.6	30.8986
41.0975646973	40.3658561707	0.731709	0.5	68.3332
36.4634170532	34.1463432312	2.317074	2.0	86.3158
61.3414649963	60.4878082275	0.853657	1.3	152.2860
70.6097564697	66.4634170532	4.146339	2.1	50.6471
53.2926864624	51.8292694092	1.463417	1.2	81.9999

100	83.65853882	16.34146	7.5	45.8955
80.48780823	73.29268646	7.195122	3.6	50.0339
57.92683029	56.09756088	1.829269	1.7	92.9333
50.12195587	48.4146347	1.707321	1.7	99.5712
98.53659058	96.58537292	1.951218	1.2	61.5005
72.92683411	60.00000381	12.92683	7.7	59.5660
57.56097794	48.29268646	9.268291	7.6	82.0000
62.68292999	59.63415146	3.048779	2.0	65.6003
58.04878616	50.9756088257	7.073177	2.6	36.7586
90.85366058	89.8780593872	0.975601	0.8	82.0007
59.14634323	58.41463852	0.731705	1.1	150.3338

Approximate range can be calculated by seeing how many %, the SOC falls by the distance travelled in that time although anomalies such as idling at the school will be omitted for now

From the 'Approx range' column, we can see the max range (driving most efficiently) the vehicle could perform over this month is 137 miles, and the minimum (least efficient driving), the vehicle could only drive 30 miles, and an average range is 80.3326 miles.

7.10 API implementation

Steps to get connected and retrieve data:

1. Install node packages using "npm install"
2. Setup .env file with your Tesla.com username (email) and password by running "cp .env.example .env"
3. Fill in your email and password to authenticate against Tesla servers, Tesla will give a token that identifies you
4. After running through the setup, run "npm run dev" To run this locally.

7.11 Myriad – Computer Cluster Setup

Logging in:

1. Open PuTTY
2. Load saved session called Socrates, which loads host name 'myriad.rc.ucl.ac.uk'
3. Log in to myriad cluster using UCL login and password

Transfer data onto system:

1. Open graphical client WinSCP
2. Select new site on left tab
3. File protocol: SCP
4. Host name: myriad.rc.ucl.ac.uk, user name/password: UCL user name + password
5. Click save and save as: myriad_house_data

6. Login
7. From desktop (left side), drag files into: stuff…/scratch/workspace/ directory (on right side) which should have transferred data to the cluster

Submitting job to scheduler:

1. Using a text editor (notepad), copy the serial example jobscript and saved it as transfer_file.sh
2. Drag the file into the scratch/home directory on WinSCP
3. Type in PuTTY: 'qsub transfer_file.sh' and the serial job should be submitted
4. Type qstat to see if it is in the queue
5. Type qexplain (ID) to read any errors that come up

7.12 Logged user drive cycle data using CANedge2 for vehicle speed

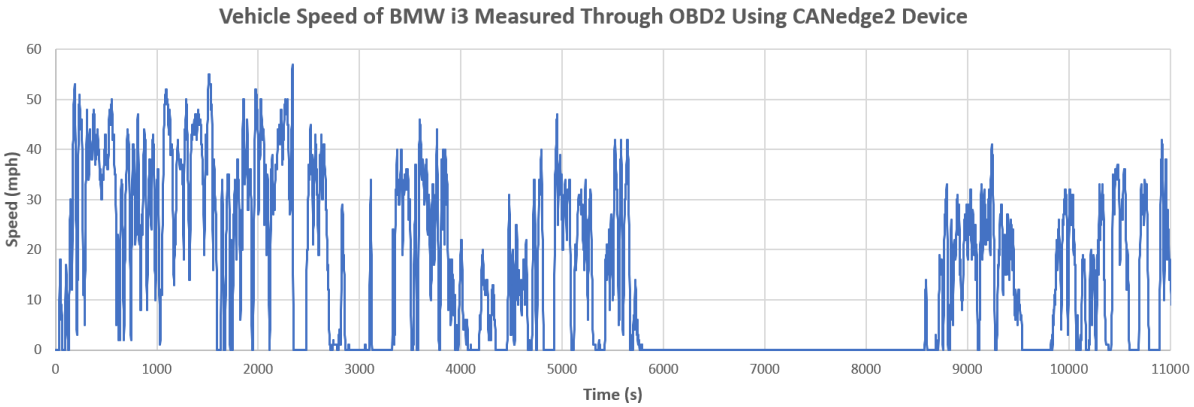

Figure 10 – The extracted data using the CANedge2, plotted showing vehicle speed vs time

7.13 Household Load Profile – Longer Time Period

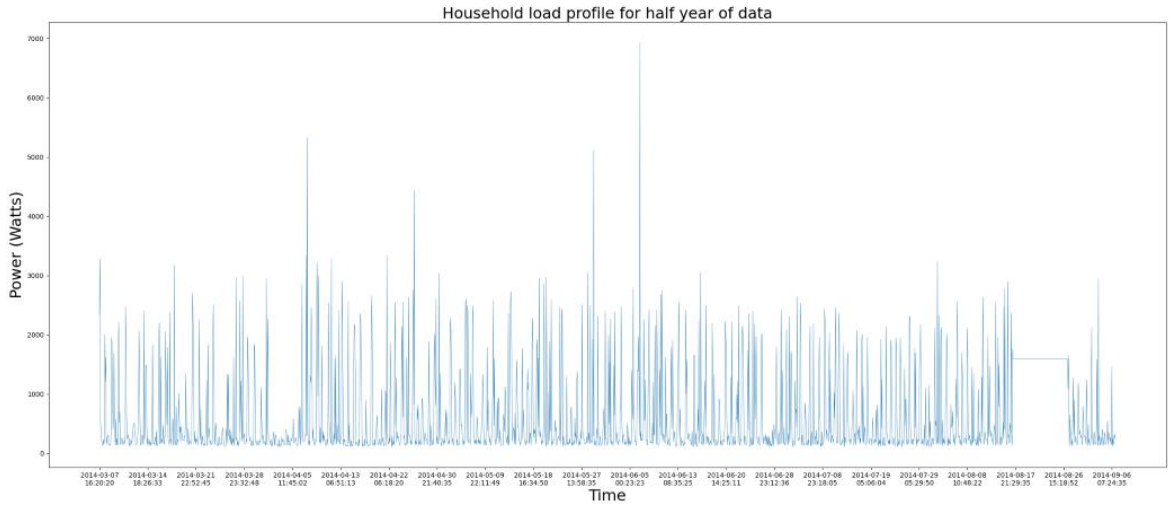

Figure 11 - Household load profile for house number 1 for half a year. Plotted using Myriad

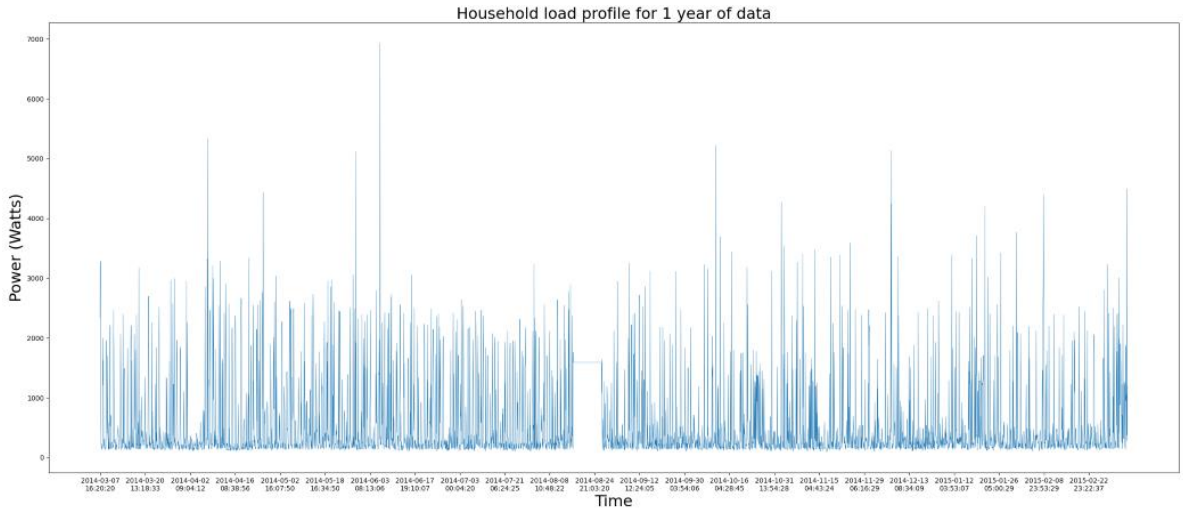

Figure 22 - Household load profile for house number 1 for a full year. Plotted using Myriad

7.14 Baseline Models Mathematical Background

Mean

$$\hat{y}_{T+h|T} = \bar{y} = \frac{y_1 + \cdots y_T}{T}$$

The $\hat{y}_{T+h|T}$ notation means the estimate for y_{T+h} $based\ on\ the\ data\ y_1, \ldots, y_T$

Naïve

$$\hat{y}_{T+h|T} = y_T$$

Seasonal Naive

$$\hat{y}_{T+h|T} = y_{T+h\_-m(k+1)}$$

Where m is the seasonal period, k is the integer part of $\frac{h-1}{m}$ (the number of complete periods in the forecast period prior to time $T+h$) [103].

7.15 ARIMA - ACF/PCF, AIC

ARIMA has 2 parameters that can be tuned using these plot, auto-regressive (AR), Moving average (MA). Using ACF/PACF plots, we can visually inspect the autocorrelation.

From figures 13,14, we are looking for values outside of the shaded region. Values inside mean it isn't statistically significant.

The ACF (autocorrelation) plot, shows the correlation of a sequence with itself lagged by some value of units. X axis shows lag number, Y axis shows the correlation sequence ranging from -1 to 1. A lag of 0 will always have correlation of 1 because the 2 series are identical. Apart from this trivial example, we look for other lag values that are significant and have strong impact. In Figure 13, ACF plot, it has significance up to value 5, therefore an MA term of up to 5 should be used.

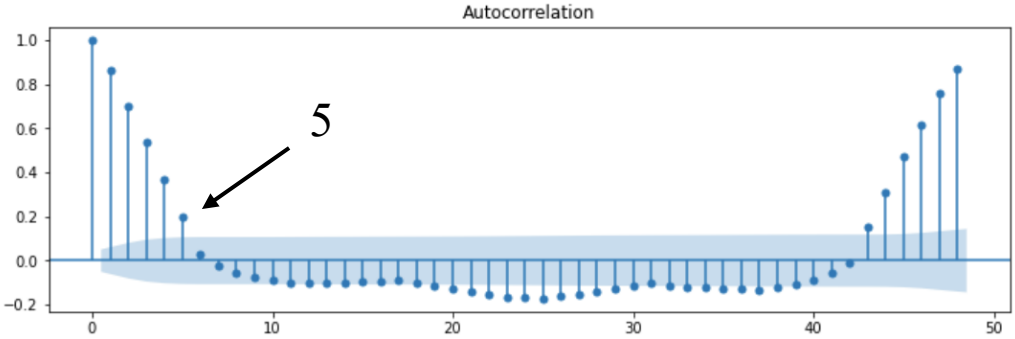

Figure 13 – ACF plot for TOU data

With PACF (Partial autocorrelation), it also shows correlation of a sequence with itself lagged by some number of time units, however only a direct effect is shown and all intermediary effects are removed meaning it tells the direct relationship between a lag value and current.

In Figure 14, PACF plot, there exists quite a few significant correlations, therefore an AR up to value 8 and also values 43, 45-48 could be potential.

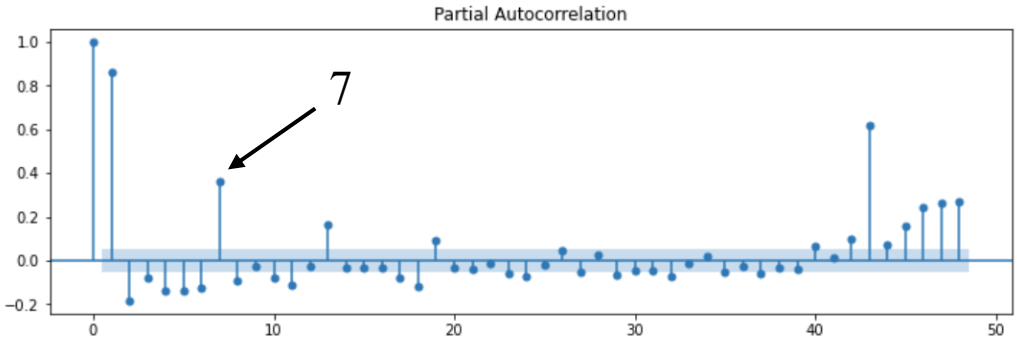

Figure 14 – PACF plot for TOU data

Also mentioned that can help determine optimal hyperparameters for time series forecasting is AIC (Akaike Information Criterion).

$$AIC = -2logL + 2k$$

Where L is the maximum likelihood, K is the number of independent variables used (K=p+d+q+1), the lower the AIC the better. A function, "auto_arima" helps find which configuration is best (Figure 15).

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=6844.663, Time=3.96 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=9240.444, Time=0.13 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=7018.582, Time=0.35 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=7885.044, Time=0.61 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=11417.375, Time=0.06 sec
 ARIMA(1,0,2)(0,0,0)[0] intercept   : AIC=6887.567, Time=1.19 sec
 ARIMA(2,0,1)(0,0,0)[0] intercept   : AIC=6842.959, Time=1.31 sec
 ARIMA(1,0,1)(0,0,0)[0] intercept   : AIC=6912.434, Time=0.77 sec
 ARIMA(2,0,0)(0,0,0)[0] intercept   : AIC=6878.756, Time=0.55 sec
 ARIMA(3,0,1)(0,0,0)[0] intercept   : AIC=6844.770, Time=2.16 sec
 ARIMA(3,0,0)(0,0,0)[0] intercept   : AIC=6856.104, Time=0.90 sec
 ARIMA(3,0,2)(0,0,0)[0] intercept   : AIC=6831.093, Time=3.39 sec
 ARIMA(4,0,2)(0,0,0)[0] intercept   : AIC=6770.306, Time=4.42 sec
 ARIMA(4,0,1)(0,0,0)[0] intercept   : AIC=6799.920, Time=2.33 sec
 ARIMA(5,0,2)(0,0,0)[0] intercept   : AIC=6747.585, Time=3.58 sec
 ARIMA(5,0,1)(0,0,0)[0] intercept   : AIC=6801.460, Time=2.95 sec
 ARIMA(6,0,2)(0,0,0)[0] intercept   : AIC=6669.508, Time=4.44 sec
 ARIMA(6,0,1)(0,0,0)[0] intercept   : AIC=6766.925, Time=2.35 sec
 ARIMA(7,0,2)(0,0,0)[0] intercept   : AIC=6624.994, Time=3.09 sec
 ARIMA(7,0,1)(0,0,0)[0] intercept   : AIC=6643.137, Time=2.41 sec
 ARIMA(8,0,2)(0,0,0)[0] intercept   : AIC=6597.091, Time=5.65 sec
 ARIMA(8,0,1)(0,0,0)[0] intercept   : AIC=6599.113, Time=4.17 sec
 ARIMA(9,0,2)(0,0,0)[0] intercept   : AIC=6601.684, Time=7.02 sec
 ARIMA(8,0,3)(0,0,0)[0] intercept   : AIC=6554.547, Time=6.95 sec
 ARIMA(7,0,3)(0,0,0)[0] intercept   : AIC=6614.091, Time=5.66 sec
 ARIMA(9,0,3)(0,0,0)[0] intercept   : AIC=6561.174, Time=7.36 sec
 ARIMA(8,0,4)(0,0,0)[0] intercept   : AIC=6534.765, Time=6.26 sec
 ARIMA(7,0,4)(0,0,0)[0] intercept   : AIC=6565.507, Time=7.72 sec
 ARIMA(9,0,4)(0,0,0)[0] intercept   : AIC=6535.388, Time=7.89 sec
 ARIMA(8,0,5)(0,0,0)[0] intercept   : AIC=6507.565, Time=6.77 sec
 ARIMA(7,0,5)(0,0,0)[0] intercept   : AIC=6494.129, Time=6.56 sec
 ARIMA(6,0,5)(0,0,0)[0] intercept   : AIC=6491.773, Time=5.41 sec
 ARIMA(5,0,5)(0,0,0)[0] intercept   : AIC=6489.034, Time=5.13 sec
 ARIMA(4,0,5)(0,0,0)[0] intercept   : AIC=6488.182, Time=4.69 sec
 ARIMA(3,0,5)(0,0,0)[0] intercept   : AIC=6488.292, Time=4.23 sec
 ARIMA(4,0,4)(0,0,0)[0] intercept   : AIC=6676.656, Time=3.91 sec
 ARIMA(3,0,4)(0,0,0)[0] intercept   : AIC=6669.008, Time=3.70 sec
 ARIMA(5,0,4)(0,0,0)[0] intercept   : AIC=6657.803, Time=4.71 sec
 ARIMA(4,0,5)(0,0,0)[0]             : AIC=6799.420, Time=2.71 sec

Best model:  ARIMA(4,0,5)(0,0,0)[0] intercept
```

Figure 15 – Auto_arima performing stepwise search for AIC to find optimal ARIMA hyperparameters

7.16 FBProphet Mathematical Background

Prophet relies on a Fourier series, to create the model simulating periodic effects [104].

It is an additive model in its simplest form:

$$y(t) = g(t) + s(t) + h(t) + e(t)$$

Where g(t) is trend (growth over time), s(t) is seasonality (daily, monthly, yearly), h(t) is holidays (predictable irregularities in the data) and e(t) is idiosyncratic changes (noise).

Changing hyperparameters will change these terms mentioned as they each effect how trend, seasonality is realised by prophet.

7.17 Classifier Module

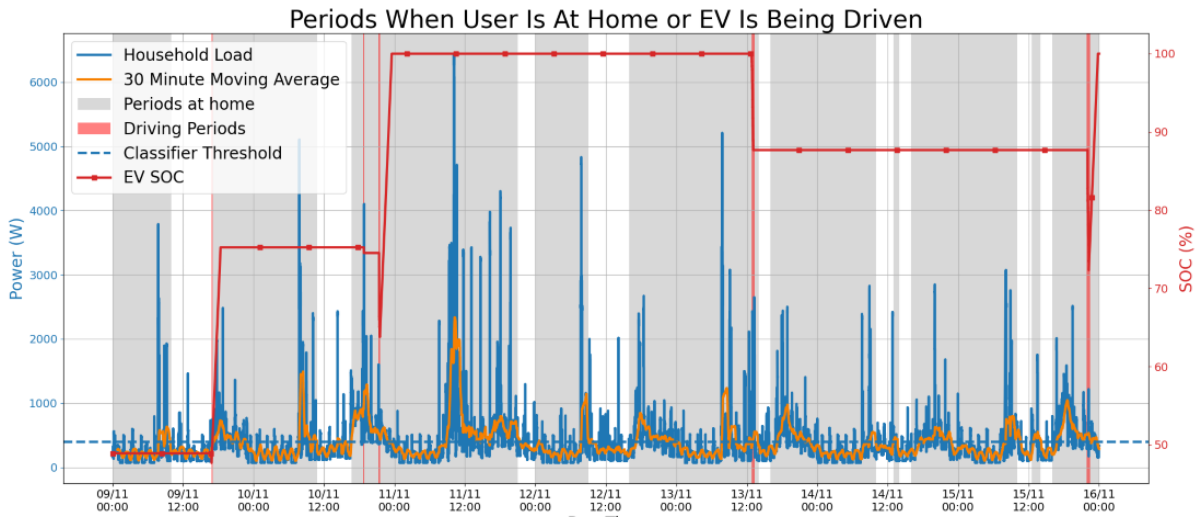

Figure 16 – Classifier module for 1 week of data

7.18 Cost comparison

Table 3: Costs for smart vs dumb charging and cost saved for 1 week of data

	9/11	10/11	11/11	12/11	13/11	14/11	15/11
Dumb Charging Cost (£)	£3.10	£1.25	-	-	-	£0.43	£0.43
Smart Charging Cost (£)	£0.82	£1.12	-	-	-	£0.39	£0.41
Difference Cost (£)	£2.28	£0.13	-	-	-	£0.04	£0.02
Percentage difference	74	10	-	-	-	9%	5%

7.19 Reverse Engineering EV PIDs

The most common protocol for reverse engineering PIDs is through CAN snipping/hacking, which can be done using a CAN streaming device such as a CLX000 [105]. This device can livestream CAN data onto a laptop via software such as Wireshark [106].

Once the data starts streaming, begin to charge the EV and monitor which PIDs and CAN data are changing. The idea is that by fluctuating a particular component of a vehicle and monitoring which PIDs change. Individual PIDs can be isolated as patterns emerge over time for example,

a set of data which continually increments over time may come from increasing SOC percentage as a car is being recharged.

7.20 Connecting the CANedge2 device to cloud servers

A function of the CANedge2 that was theorised but not fully implemented due to time and access constraints, was connecting the CANedge2 to the cloud. The CANedge2 device has Wi-Fi access points that allow the logged data to be uploaded over-the-air to a private server [66][67].

This can be achieved by configuring the config file to connect to a 3G/4G hotspot or Wi-Fi access point, then synchronising it to a server such as Amazon Web Services (AWS). This has been completed and the config file can be found on this projects GitHub page [87].