Comparative Analysis of Machine Learning Techniques for an Hour-Ahead Forecasting of Electric Vehicle States

by

Mahmoud Kiasari

Submitted in partial fulfillment of the requirements for
the degree of Master of Applied Science

at

Dalhousie University
Halifax, Nova Scotia
Aug 2023

My beloved parents, whose unwavering love and support have been the foundation of my academic journey, your sacrifices, guidance, and belief in me have shaped the person I am today. Thank you for your love and passion towards my ambitions and goals.

Moreover, to my beloved wife, Elaheh, none of these works would be possible without her. You have been my rock and my greatest source of strength: your enduring support, genuine understanding, and unwavering encouragement. Through the remarkable journey, your unwavering belief in me and constant presence have been an invaluable wellspring of strength and motivation, guiding me through ups and downs. Words fail to capture the depth of appreciation I hold for your steadfast support and unwavering companionship.

May this thesis serve as a tribute to the love and support of my parents and wife, who have been instrumental in my success.

Table of Contents

List of Tables

List of Figures

Abstract

This thesis presents a method for forecasting the state of electric vehicle (EV) batteries (Vehicle-to-grid, Grid-to-vehicle) for the next hour towards leveraging updated data as the grid's auxiliary power source. The aim is to optimize grid power utilization and mitigate concerns such as voltage and frequency variations, power loss, and harmonic distortion. The proposed prediction model incorporates various distributed energy resources, including demand-side management strategies and energy storage devices, to support the grid. Multiple machine learning (ML) techniques, including logistic regression, artificial neural networks (ANN), naive Bayes, K-nearest neighbour (KNN), and Support Vector Machines (SVM), are employed to predict V2G and G2V conditions. The model is refined by including vital features significantly influencing the outcomes and applying feature selection techniques. Furthermore, correlation time and Gaussian correlated noise are employed to assess feature dependency and evaluate the methods' robustness against noise. The accuracy analysis revealed that the ANN technique performed best among the tested ML techniques. It demonstrated superior accuracy in predicting the state of EV batteries. Additionally, feature selection, correlation time, and Gaussian correlated noise enhance the model's performance and evaluate its robustness.

List of Abbreviations Used

ACC	Accuracy
AC	Alternative Current
AI	Artificial Intelligence
AMI	Advanced Metering Infrastructure
ANN	Artificial Neural Network
API	Application Programming Interface
BEV	Battery Electric Vehicle
BEVs	Battery Electric Vehicles
DC	Direct Current
DER	Distributed Energy Resources
DSO	Distribution System Operator
EM	Electromagnetic
EMS	Energy Management System
EV	Electric Vehicle
EVCS	Electric Vehicle Charging Stations
EVSE	Electric Vehicle Supply Equipment
FCEVs	Fuel-Cell Electric Vehicles
FN	False Negative
FP	False Positives
G2V	Grid To Vehicle
GHG	Greenhouse Gas
HEVs	Hybrid Electric Vehicles
ICE	Internal Combustion Engine
KNN	K-Nearest Neighbor
ML	Machine Learning
MSE	Mean Squared Error
NREL	Energy's National Renewable Energy Laboratory
OBC	On-Board Charging
PHEV	Plug-In Hybrid Vehicles
PV	Photovoltaic
RE	Renewable Energy
ReLU	Rectified Linear Unit
RPM	Revolution Per Minute
SGD	Stochastic Gradient Descent
SOC	State Of Charge
STD	Standard Deviations
SVM	Support Vector Machines
Tanh	Hyperbolic Tangent
TN	True Negative
TP	True Positives

V2B	Vehicle To Building
V2G	Vehicle To Grid
V2X	Vehicle-To-Everything

Acknowledgments

I feel towards these remarkable people:

First and foremost, my co-supervisor, Dr Aly, deserves special mention. His unwavering support, heartfelt encouragement, and sincere guidance have been the driving force behind my work. He has been there for me every step of the way, offering invaluable insights and motivating me to aim for greatness.

I am truly indebted to my supervisor, Dr Little, for his immense knowledge, experience, and wisdom. His mentorship has been invaluable, helping me navigate the complexities of my research and gain a deeper understanding of the subject matter. His keen eye for detail has illuminated new perspectives and improved the quality of my work.

Dr Kember, too, deserves my heartfelt appreciation for his contribution. His thoughtful comments and thorough understanding of my research have truly elevated the significance of my findings. His belief in my abilities has boosted my confidence and pushed me to push the boundaries of my research.

I cannot forget to express my gratitude to Dr Gu, a member of my committee, whose feedback and suggestions have been instrumental in shaping my work. His unique perspectives and expertise have broadened my horizons and enhanced the overall impact of my research.

Chapter 1 : Introduction

1.1 EV Charging Problems

Clean and efficient electric energy generation, transmission, and consumption are now required to offset the consequences of global warming. Because transportation-related greenhouse gas (GHG) emissions account for roughly 27% of total US GHG emissions, it is the leading source of US GHG emissions[1]. Between 1990 and 2020, GHG emissions in the transportation industry increased faster than in any other sector. The development of renewable energy(RE) generation, energy storage, and EVs are among the critical stages toward cleaner and more efficient electricity and transportation sectors. With the growing number of EVs, the interplay between the electric automobile and the energy grid is becoming increasingly apparent.

However, the benefits of injecting EVs into the grid are one side of the subject. The amount of load EVs could place on the grid due to charging needs is another area researchers are concerned about. According to this paper[2], there are some flaws in using EVs in an unexpected way of charging. A drop in voltage value, an increase in the level of unbalancing load, an increase in the overloading of power system components like cables or transformers, an increase in power losses, an increase in harmonic distortion, and deterioration of voltage and current transients during faulty conditions are all possible consequences of EVs being charged at random. Various researchers have proposed charging management methods for the EV fleet to mitigate this problem. To mention some, this paper[3] offered a machine-learning way to predict each EV battery condition, and this work[4] proposes tackling the problem using a Real-Time Digital Simulator. Even though much excellent work has been done up to this point, it still has flaws. Neglecting to account for the sheer volume of EVs on the road, failing to design a system that is adaptable enough to accommodate diverse energy sources like renewables, , and failing to implement adequate measures to ensure that each EV's battery is used to its full potential despite differences in size, all contribute to this dire situation.

Furthermore, standardization and interoperability of charging protocols among different EV models and charging infrastructure are essential. This ensures efficient charging processes and provides convenience for EV owners. The scalability of charging infrastructure must also be considered to meet the growing demand for EVs. Comprehensive infrastructure planning is necessary to anticipate future needs and ensure the availability of sufficient charging stations.

Moreover, grid reinforcement and optimization should be prioritized to accommodate the increased load from EV charging. This entails grid upgrades, distribution system planning, and the implementation of advanced grid management techniques. By addressing these aspects, potential challenges such as voltage drop, load unbalancing, and power losses can be mitigated effectively.

1.2 Previously Proposed Models For Fleet Management of EVs

Numerous studies and investigations have been carried out to find a solution to the management issues. The MATLAB software implementation paradigm, seen in these articles **[5]–[9]**, was the most often used method for working in this field's most famous area. These authors have aimed to provide a standardized model for fleet management to optimize consumption and generation, which might help minimize the issues that have been occurring up until now. Also, using the data from these models, they have made various conclusions according to the prediction. It is well-known that earlier works accomplished a great deal in management. A few overlooked points in their works may be brought up.

There has been a significant amount of research recently on using ML and advanced metering infrastructure (AMI) for predicting EV charge scheduling. For example, in one study**[10]**, SVM was used to analyze home charge scheduling based on user energy consumption and EV state of charge (SOC) data. Another paper **[3]** used ANN to determine the optimal schedule for EV charging, considering factors such as household power demand and EV energy demand. These works demonstrate

ML and AMI's potential for improving EV integration into the power grid by enabling real-time decision-making about charge scheduling. Paper**[11]** also presents a model-free reinforcement learning technique for scheduling EV charging in the context of V2G systems. This technique coordinates charging safely and efficiently while considering real-time prices, commuting behaviour, and energy needs. Other related work about using different ML methods also have been done in this field. **[12]** applied supervised machine learning models to forecast energy consumption at Electric Vehicle Charging Stations (EVCS) in Indonesia, using historical charging transaction data and additional weather condition data as input. Four machine learning algorithms were evaluated: Random Forest, Support Vector Regression, XGBoost, and Multilayer Perceptron. Additionally, this paper **[13]** conducted a comparative analysis to determine the most suitable machine-learning technique for predicting the SOC of EVs based on vehicle flow data from the city of Agartala. Three machine learning techniques, namely CG-SVM, bagged tree regressor, and boosted tree regressor, were evaluated using performance metrics such as mean squared error, mean absolute error, R-squared value, and root mean squared error.

Battery degradation has always been a massive concern in EV management, like this study**[13]**, which talks about the degradation of batteries during V2G condition usage of the battery. Reasons like this have increased the importance of accurate prediction for EV usage. While this matter showed its extraordinary vitality, a few related works have been considered a proper way to predict by considering essential features like battery voltage, current, or temperature.

In this work, it becomes evident that previous approaches in the field of EV battery prediction suffered from significant limitations. One critical issue is the lack of consideration for alternative methodologies and the untapped potential of various machine-learning techniques. Instead, this research takes a bold step forward by exploring not just one but five distinct machine-learning techniques: logistic regression, ANN, naive Bayes, KNN, and SVM. This diversified approach surpasses the boundaries of traditional prediction models and provides a deeper understanding of EV battery behaviour. Additionally, the importance of feature selection, which was overlooked in prior studies, is recognized in this work. The predictions achieve higher accuracy and reliability by focusing on the most influential factors while disregarding less consequential ones. Moreover, this work goes beyond the norm by incorporating correlation time and correlated noises, factors previously ignored. These comprehensive enhancements expose the shortcomings of previous research, positioning this work as a novel contribution that revolutionizes the field of EV battery prediction.

1.3 Scope of Research

This thesis aims to address the limitation of previous works in EV fleet management using ML models that consider a wide range of features and factors that affect the battery behaviours and EV conditions on the grid. Moreover, alongside the vast amount of data with various features, different ML algorithms are used to select the best model for the prediction. Meanwhile, the feature selection method is applied to decrease the least affected features to increase the data alongside the processing time decrement. Alongside the consideration of Correlation Time, to address different feature dependencies with Gaussian Correlated Noises, to evaluate the power of each method against real-life noises. These concepts are entirely elaborated on in this thesis.

This work has a fleet of four EV batteries and four houses as a load demand. It can be adjusted for a higher number of the fleet as well. Information about four houses using this dataset**[14]** uses twenty-four hours a day to input power consumption. An important fact about this data is that the period of each house's power consumption is minute by minute, and based on the fact that this thesis is focused on the next-hour prediction, the amount of data gathered by using the input power consumption should be sufficient. That is why more than two months of a year are considered for this work. These days are randomly selected from various months to generalize the prediction model. Also, other features are added to this model to modify it for a better prediction resolution. Using this data will help the system adjust itself in various situations and decide whether each battery should act as a source, load, or idle one. The important part about the data that is used for this model is that, at the first point, raw data from an open source site was gathered, and after that, by declaring different scenarios about the thresholds of EV's battery storages, time of the day, and power consumption limitation the data got modified to be proper enough for the ML usage.

1.4 Methodologies and Contributions

Researchers in forecasting have been motivated to use this cutting-edge technology for precise forecasting due to ML's ability to learn across diverse subject areas. The number of possibilities for developing an accurate prediction model is quite broad. On the other hand, Python is selected as the software used in this endeavour because of its simplicity and ease of reading its code. In this study, several comparisons among the many accessible approaches for model development are employed, like Accuracy, F1-Score, and Mean Squared Error(MSE) as the ways of evaluation. These comparisons aim to improve the accuracy of the suggested models in Python. This study examined comprehensively to determine the most accurate machine-learning technique for a specific prediction model. To this end, various methods are employed to mitigate potential biases or uncertainties that may have arisen during the analysis.

In addition, Correlation Time is employed to increase the certainty of the impact of each feature on the overall prediction concept and to understand the utilization of interconnections. Furthermore, this prediction model must demonstrate its practicality in real-life situations where uncertainty and non-ideal conditions are likely to occur. To evaluate the functionality of each model under such circumstances, the presence of noise becomes vital. Therefore, another significant contribution of this work is the utilization of Gaussian Correlated Noises to address this matter. A study needs a multidimensional approach, so different strengths for correlation time and noise are considered. This approach allows to showcase the power and effectiveness of the proposed work. The final objective of this thesis is to identify the optimal technique for handling diverse inputs and to demonstrate its efficacy through a thorough examination of its performance.

1.5 Thesis Outline

The thesis is structured into six main chapters to address the research objectives systematically. Chapter 1 provides an introduction, highlighting the EV charging problems and the existing models for fleet management. The scope of the research is outlined, followed by a description of the method contribution. This chapter concludes with an overview of the thesis outline. Chapter 2 delves into the Energy Management System, covering various aspects such as different EV types, their advantages and disadvantages, EV charge port, conductive and inductive charging, and the classification of battery electric vehicles (BEV) based on energy transfer.

Additionally, the chapter explores off-board and onboard charging, the role of the distribution system operator, battery energy management systems, and the concept of unidirectional and bidirectional charging. The configuration of a DC fast charging station for V2G/G2V is also discussed, along with the components involved. Chapter 3 focuses on Machine Learning, introducing different types of ML learning, including unsupervised and supervised learning. This chapter explores regression techniques such as logarithmic regression and other ML algorithms like Naïve Bayes, K-nearest neighbours, Support Vector Machines, and Artificial Neural Networks. The chapter concludes with a discussion on classification evaluation and feature selection methods. This chapter also covers the implementation of the ML model. Chapter 4 presents the results of the research. Machine learning results for different algorithms are discussed, including Artificial Neural Networks, Logistic Regression, Naïve Bayes, K-nearest neighbours, and Support Vector Machines. It addresses feature selection and correlated noise. Chapter 4 will discuss the prediction of EV state based on the usage of Gridlab software, which will be explained in other chapters. Chapter 5 provides the results, summarizes the key findings of the research, and suggests future works to enhance the field of EV battery prediction further. Finally, Chapter 6 will talk about the conclusions alongside future works.

Chapter 2 : Energy Management System

2.1 Introduction To Energy Management System

RE sources, especially wind and photovoltaic (PV), face challenges related to unpredictable generation. Similarly, other technologies aimed at addressing daily challenges also encounter similar issues. Despite the modern prediction equipment, too much reliability of REs, like the wind to the weather, raises additional problems for affluent countries seeking to shift their power output to renewables. Another issue is the high cost of the first equipment required to increase the budgets to ensure their existence at proper times. To some extent, EV batteries can solve some of these issues**[15]**. EV batteries are used as a storage system that can manipulate mentioned challenges. In today's context, as Vehicle-to-everything (V2X) technology gains traction, companies are exploring using used batteries to store energy and serve as backup sources for renewables. While this presents an opportunity for enhanced energy storage capacity, it also introduces challenges for the grid**[16]**. Integrating and managing EVs or any RE connection to the grid poses additional complexities. These include voltage and frequency variations, which can disrupt the grid's stability. Moreover, there is the risk of overloading the system due to the increased demands placed on it by the growing number of EVs and RE sources**[17]**. Overcoming these drawbacks and effectively managing the interplay between V2X, used batteries, and the grid is crucial for successfully integrating sustainable energy solutions.

The first generation of EV charging focused only on injecting energy from the grid into the EV. This one-way connection is referred to as unidirectional charging. In a subsequent form of charging, where the EV may also operate as a power source, injecting electricity into the grid was proposed. This type of charging is known as bi-directional charging**[18]**. EV batteries may be employed as an additional source for the grid with the aid of this innovative technology. In addition, an intelligent system is vital to make decisions regarding the state of each battery (Charge, Discharge, or Idle). This system's primary role is to manage EVs to operate as a source or load when other elements like power usage, time of day, and other sources are present. Recent papers **[5]–[9]** proposed a bi-directional EV charging structure. The control portion of these models is crucial since it assists the system in determining its purpose for real-time prediction.

Additionally, since there are infinite participants in the fleet, such as homes and vehicles, the system should be able to adapt to any situation. In recent years, MATLAB has assisted manufacturers and researchers in overcoming these types of difficulties and related issues. By executing the proper structure of parts in that program, engineers may ensure that their project can be implemented appropriately for its utilization in most circumstances. Additionally, reducing mistakes and boosting efficiency are considered. Another factor that any system should consider is the capacity to forecast future events. Electrical engineering's most potent tool, ML, is frequently applied in many facets of management. Not only does prediction utilizing this new technology reduce any errors a system may make, but it also enables the grid's generating component to adapt to impending consumption conditions.

Engineers were forced to create a model after experiencing difficulties with time management in real-time and one-day-in-advance forecast procedures. This model might be crucial in the management component of the fleet supervision system. Also, this technology should be able to do its job without attention to the increasing size of input elements connected to it. Instead, the management should be able to persuade each EV owner to be happy with their contribution as an auxiliary input to the system. Instead of using a real-time forecast of each EV battery's status as a backup in the real-time mode of operation, hour-ahead prediction is another essential component for the grid and generation portion. It is the responsibility of ML to develop a model that would make use of real-time data to assist management in rearranging its operational components in a way that would satisfy the energy requirements of the grid to compensate for the lack of power, particularly during peak hours of the day, and avoid any possibility of a blackout. The efficiency and adaptability of this developed model should be considered to respond appropriately to input elements. By combining these two systems, it would be possible to accurately estimate future day's demands for generation and fleet planning in addition to real-time handling management of EV batteries as an extra hand for the grid.

Given these observations, the findings **[19]**suggested an energy management system (EMS) to effectively coordinate EV units' integration within the microgrid. Specifically, an EMS would manage peak demand by scheduling and controlling EV charging and discharging that aligns with the microgrid's overall needs. It is important to highlight that a microgrid refers to a small-scale power grid capable of functioning independently or in conjunction with other microgrids. These microgrids are characterized by their decentralized nature and can be observed in diverse settings, including districts or integrated within energy generation facilities. Consequently, most EV applications within microgrids are expected to occur within the residential sector, focusing on fulfilling households' energy requirements.

2.2 Different EV Types

Battery Electric Vehicles (BEVs), Hybrid Electric Vehicles (HEVs), Plug-in Hybrid Vehicles (PHEVs), and Fuel-cell Electric Vehicles (FCEVs) are considered different types of EVs[20]. In BEVs, the primary power source is only batteries that the charging points can charge. A hybrid electric vehicle merges an internal combustion engine with an electric power system. Integrating an electric powertrain is intended to achieve improved fuel efficiency or enhanced performance compared to a conventional car. In HEV cars, there are two modes of operation: the battery and the ICE(Internal Combustion Engine) motor. One of them is parallel, and the other one is in a series. The first mode involves the battery and motor operating independently as the primary power source during different instances. In the second mode, they synergise to enhance the car's performance. However, a drawback of these vehicles is their inability to be charged directly by connecting them to a charging point.

Regenerative braking is the charging way for them. Regenerative braking is a method of slowing down an electric vehicle by converting mechanical energy to electrical energy through the generator function of its motor while also generating brake torque and storing it in storage devices**[21]**. PHEVs are the modified version of HEVs that drivers can plug into the charging unit. FCEV works with Hydrogen, and like ICEs, they can be fuelled. The only difference is that they will be filled with Hydrogen instead of gasoline.

2.3 EVs Advantages and Disadvantages

Oil (petroleum), coal, and natural gas (gases, fuel), formed from the remnants of extinct plants and animals, serve as the traditional energy sources for internal combustion engines (ICEs), which inevitably contribute to greenhouse gas emissions and pose environmental risks. In power plants and hybrid electric vehicles, fossil fuels are burned to generate electricity for recharging the battery. Managing air pollutants at a power plant proves more feasible than tackling emissions from the exhaust pipes of millions of gasoline-powered vehicles. Hybrid electric vehicles incorporate electronic controls to optimize engine usage, running them only when necessary and with greater efficiency. This approach to electricity consumption leads to enhanced air quality, particularly in major cities, which has become a paramount concern for the nation. Compared to fossil fuels—natural gas, coal, and oil—derived from the remains of long-extinct plants and animals, a cleaner alternative exists for powering automobiles.

In recent times, EVs have made significant strides in addressing the issue of limited range. However, the cost of EVs remains a substantial challenge for potential owners. Furthermore, certain features in EVs, such as air conditioning, consume energy and can deplete the battery at an accelerated rate. Another hurdle in the widespread adoption of EVs is the availability of charging stations. Due to EV batteries' limited energy storage capacity, the overall power and driving range is compromised. Additionally, the time required for a full EV charge still falls behind the refuelling time of conventional cars, presenting a disadvantage in terms of convenience and usability. Despite advancements, these factors contribute to the ongoing competition between EVs and ICEs in the automotive market. **[17]**

2.4 EV Charging Port

The battery pack of an electric car can receive electricity from an external power source through the charging connector. Electric vehicle supply equipment (EVSE), more often known as a charging station, is the name given to these power sources. Charging occurs when an EVSE is inserted into an EV's charge port at a public or private charging station. The speed of EV charging is related to the power that the charging port will be injected into the car. Level 1, Level 2, and Level 3 are three different levels now. The lowest power level of charging, known as Level-1 charging, is the slowest and typically occurs during nighttime, primarily in residential settings. The input voltage used by an Alternative Current (AC) charger is 120V or 230V and transports around 1.92kW of electricity. Level-2 AC chargers are typically used in business locations like malls or offices, and they can produce power up to 20k by accepting 208Vac or 240Vac as input voltage. On-board Level -1 and Level-2 ac chargers have a restricted power rating and a more extended charging period, which has sparked the development of Level-3 Direct Current (DC) fast chargers that can handle power from 50kW to 300kW. They can deliver dc power of up to 800V, roughly 300V, and charge the current EV batteries in 30 minutes. Since, In Levels 1 and 2, the input voltage connected to the EV is AC and needs to be converted to DC, these EV types should have one part named the On-Board Charging (OBC) part. This part is responsible for converting the AC to DC or vice versa. This part is mitigated in Level 3 charging units since the charging port has an Off-Board Charging unit that delivers energy to the car. Table 1 depicts the difference between charging levels.**[22]**.

Table 1 Charging station classification based on charging power level

Charging station	**Charging location**	**Power supply**	**Charge power level**	**Charging time**
Level-1 (AC)	On-board (Residental)	120/230V	1.44kW - 1.92kW	11 - 36 hours
Level-2 (AC)	On-board (Home)	208/240V	3.1kW – 19.2k@	2 – 6 hours
Level-3 (DC Fast)	Off-board (Public)	300-600Vdc	50kW – 350kW	Less than 30 minutes

2.5 Conductive and Inductive Charging

There are two ways for EV batteries to be charged. For conductive charging, the EV connection and the charge input directly touch one another. Multiple AC and DC chargers are standard in many new cars. It is necessary to utilize an external charger to provide consumers greater flexibility in where and when they may charge their automobiles, whether at home or in public. Although conductive chargers are predominant in the current EV market, they have several disadvantages, such as the need for the user's intervention and safety issues. Inductive chargers are a reliable alternative to solve some of these drawbacks.

Another way of charging where the physical connection vanishes is called inductive charging. An electromagnetic (EM) field transfers power between two objects in an inductive charging process. Therefore, a charging station is designed specifically for this need. Inductive coupling is the transmission of energy to an electrical device. Indeed, the wireless option is the most crucial superiority of this way compared to the previous one. However, there are several difficulties associated with inductive charging. The most challenging part of wireless charging is getting the coils aligned adequately for maximum efficiency, but thanks to recent automation developments, parking assistance systems may now be included in the process**[23]**.

2.6 Off-Board And Onboard Charging

Onboard charging refers to the way of EV charging where the input that is connected to the EV is AC, and since the power needed for the battery is DC, it has a converter inside the car itself that is responsible for changing AC to DC. On the other hand, Off-board charging is where this conversion process will be done outside the card and in the station.

The main challenge lies in the power required for Level-1 charging equal to the onboard charger's maximum power capacity. Reducing onboard charger usage is primarily driven by weight, space, and cost considerations. Resonant circuits can be employed to mitigate various adverse effects. In recent years, integrated charging systems have become standard in the industry. However, there is currently a surge in demand for off-board fast-charging stations. Although DC stations have a higher price tag, they offer numerous exciting possibilities. These include reducing battery weight, providing multiple high-power charging options, enabling faster charging, implementing improved battery control mechanisms (such as addressing heating issues), and facilitating seamless communication between utility companies, organizations, and commercial location owners to establish favourable billing arrangements and contractual opportunities**[24]**.

As mentioned before, there are various EVs; however, in this subject, the only focus is on BEVs, whose power sources are batteries. Also, the assumption is that off-board charging is the primary concern in the current matter. Three-phase ac buses run between 250V and 480V line-to-line voltage for an AC-connected system. Because each charger unit includes an AC-DC rectifier and a DC-DC converter, there are more power stages, which raises the cost and complexity. However, because power electronics and ac power distribution systems are well-equipped and advanced, most fast and ultra-quick charging stations use an AC-connected system. The low-frequency transformer on the input side is linked to a central AC-DC rectifier in DC bus designs. Any variance from the grid side is avoided because of this structure's increased flexibility for the system. In DC bus structures, efficiency is higher, and control method implementation is more straightforward as the number of AC-DC rectifiers decreases. The lack of well-established protective standards for dc bus EV charging stations makes this problem show itself during different EV situation modes of usage. Due to lower installation costs and increased dependability, the second isolation method, accomplished using a high-frequency transformer in the DC-DC power stage, is preferred. The power density of the transformer is increased by operating it at a high frequency (between 50kHz and 300kHz) as opposed to the line frequency (50Hz or 60Hz). However, isolated DC-DC converter switching modulation and control can be challenging, and bidirectional operation calls for additional work.

2.7 BEVs Classifications Based On Energy Transfer

The integration of BEVs is critical for the operation of the electric grid. The integration must be based on the charging and discharging of PEVs, which must be accomplished utilizing a variety of techniques and control mechanisms. PEVS has three different applications: V2H, V2B, and V2G**[25]**

2.7.1 Vehicle to Home (V2H)

The concept that an EV is connected to the home to act as a source for loads in the house is called V2H. The primary goal of V2H is to store and transmit energy from locally distributed energy resources such as solar and wind energy to home loads. This will save energy expenses, increase efficiency, and offer backup power. V2H contains at least one BEV, a bidirectional charger, home loads, small-scale distributed generation, a smart meter, a home grid, and a home energy management system. Smart metering is particularly crucial in the V2H idea. It will store data regarding system use and offer an interface for sending and receiving information from utilities. This approach may be implemented into more extensive control systems to alleviate the strain on distributed energy networks. Also accessible is BEV users' ability to charge utility requests via their smartphones. In general, V2H will be called V2L, where L is represented as load in this expression. The load can be any household, battery, or consumer device.

2.7.2 Vehicle to Building (V2B)

This matter is like V2H with the difference that, here, the destination of injected power of EVs is bigger buildings like hospitals or shopping centres, and the power source is a fleet of EVs rather than one.

2.7.3 Vehicle to Grid (V2G)

Researchers are considering newer power production methods by increasing the number of people needing electricity daily. In recent years, using REs has become popular since they are convenient and primarily available**[26]**. Additionally, the increasing number of EVs in the last decade has made scientists believe they can use the saved energy in EV batteries as an ancillary power generator to connect to the grid. When the power demand exceeds the production of the power plant's base load, the unused energy is squandered. Furthermore, regulating the voltage and frequency of the power grid will significantly raise the power grid's running costs, which is why EV batteries are considered the way to mitigate the mentioned problems. By connecting them to the grid, or most explicitly, using their battery as an ancillary option to help the grid, the mentioned problems can be solved to a very acceptable extent. This technology is called V2G, which Kempton William offered for the first time at the University of Delaware in 1997. Three components are necessary for a successful interaction between electric cars and the grid: bidirectional power transformation, communication and control, and intelligent measurement devices **[27]**. Two main advantages that V2G has are power loss reduction and possible control of power factor**[28]**. Before evaluating this idea, other concepts should be considered.

2.7.3.1 V2G Applications

Valley filling and peak shaving are two load-balancing mechanisms made possible by the supplied power (charging during the low peak hours and discharging during high peak hours into the grid). Also, V2G may provide peak power for three to four hours daily. Peak power refers to providing energy during periods of peak demand. This alleviates strain on the electrical grid by decreasing the energy demand gap, transmission losses, and transmission loss investments. Moreover, V2G may function as a spinning reserve, providing electricity to the operator within 10 minutes of a request being made. However, the energy delivery device should be compensated more for the unexpected need for power. Capacity pricing accounts for the cost of reacting within one minute of notification, whereas energy pricing accounts for the amount of energy delivered to the grid. Only for a few hours each day can an electric vehicle act as a spinning reserve**[25]**.

2.7.3.2 V2G Advantages And Disadvantages

Since the cost of power is lower than the cost of gasoline, in the long term, EVs are less expensive than ICEs, and based on this reason, more people can buy EVs not only as a vehicle but also as an ancillary power for the grid to help the system. According to the last matter, the government has some encouraging plans to persuade people to use their EVs as a part of V2G. For instance, this article**[29]** estimated that each EV could earn 1000$ per year for its owner by using itself as a V2G. Moreover, relying on more REs like BESS of EVs can lead to using fewer fossil fuels, enhancing the global warming problem.

On the other hand, it also has some drawbacks, alongside all of the advantages that V2G has. In the long term, buying an EV could benefit the owner economically. Nevertheless, the high initial cost of EVs hinders customers from buying EVs to use them for themselves or connecting them as a V2G. In the last years, manufacturers have tried to reduce the cost of this product to make purchasing easier. Another problem related to using EVs is that since the charging/discharging cycle of an EV is higher than what is dedicated to it, the battery degradation of these cars will be a significant concern for owners. This matter intensified when I learned that EVs are so expensive because of their batteries. However, in recent years, this price has decreased dramatically, and nowadays, the owner should pay around 100 per kWh for the battery**[30]**. Also, smart EV charging controls battery degradation risks **[31]**. The availability of a charging station is another issue for EV owners. Unfortunately, not everyone can afford a property with a dedicated power outlet. Modern society's preference for apartment living presents a challenge for future EV owners: limited access to electrical outlets. Unlike other fuels, electric vehicles currently do not have a strong position in the infrastructure of charging stations. Filling EV batteries with power is time-consuming and might be another reason to prevent people from exhibiting a positive side to this topic as an alternative to utilizing gasoline or other non-RE sources as a source for the automobile. These limitations, like those of other technologies, will soon be remedied. Today, when oil prices keep rising and their environmental impact keeps growing, diversifying energy sources is more important than ever.

2.8 Distribution System Operator(DSO)

The entities in charge of distributing and controlling energy from generation sources to ultimate users are DSOs. Investments in automation, smart meters, real-time systems, big data, and data analytics are required to secure the DSO model. The DSO model employs intelligent meters, which enable bi-directional energy flow reading and real-time communication. This allows for detecting disruptions, restoring supply, and monitoring clients' daily use via digital consultation platforms**[32]**.

2.9 Battery Energy Management System

Since, based on statistics, EVs are, on average, 20 hours a day **[3]**, their battery can be a good source for helping the grid in times of need. Battery energy storage system (BESS) integration into the system is the way that V2G is mainly based. This way, EV batteries can be used as another ancillary power source connected to the grid and help the generators compensate for the power shortage between consumers and producers. The advantages of using EVs as BESS batteries are regulating power factors, compensating active/reactive power for the grid, and acting as a backup for REs[1].

2.10 Unidirectional And Bidirectional Charging

Unidirectional charging mostly refers to the normal charging stations where energy flows through EVs directly from the grid. However, an intelligent charging protocol needs to control these charging schedules since a high number of EVs charging simultaneously can put lots of pressure on the grid, which can cause a problem in voltage quality[34]. Moreover, managing the charging situation is crucial for the EV battery. Any disorder in the schedule or amount of energy that flows through the battery itself can damage the structure and increase the rate of battery degradation. Since this operation is low investment cost and highly available in public places like houses, it is more popular than the bidirectional one**[35]**.

The bidirectional charging way is another alternative for EV charging that not only has all of the features of the unidirectional one but also contains more features like load levelling, peak load shaving, reactive power support, active power regulation, and harmonic filters**[36]**, **[37]** Although it has some drawbacks,

[1] RE Source (RES) is the name of the energy that is produced by REs that can be used for the grid/load as well. The problem with this energy is that, since it is related to REs like solar, or wind, the fluctuations that exist in this energy make its usage risky. Weather condition plays a major role in this matter. As a backup for this energy, BESS of EVs can also be used to give the assurance of this energy to costumers that it can provide high quality and constant energy at every time that is available[33]

first and most important one is their battery degradation. Since, in this plan, the EV itself plays a significant role, and owners should see their benefits in it, managing the charge/discharge cycle is essential to decrease the probability of battery damage [38]. Moreover, requiring an intense connection between the power grid and the EV, power system strengthening, extensive investment in V2G infrastructure, and societal challenges are other problems that should be considered**[39]**.

2.11 Battery Charging

Lithium-ion batteries are the most common batteries these days. The typical voltage range for batteries is 320–400 volts. This lethal voltage might allow electricity to pass through a normal situation[40]. However, nowadays, manufacturers are trying to produce EVs with more battery pack output voltage. Increasing the rate of the battery pack has some advantages and disadvantages. Higher battery pack voltage of EV will lead to better Revolution Per Minute (RPM), which leads to more power being injected into the wheels. Also, in using charging points, since the relationship between power and current is inverse, in the same amount of power, using a higher voltage package leads to having lighter charging cables. However, increasing the battery's voltage can harm the people trying to work with it or cause electrocution. Also, higher voltage could cause more probable arcs in them that should be considered battery safety. Fuses, which are an essential part of EVs, also face problems. Another difficulty is that a battery's capacity (in Ah) is only as good as its weakest cell. Therefore, the more cells in a series, the more probable that the weakest one will have an influence. This is less of a concern for packs of numerous lower-capacity cells connected in parallel first, then in series, than for packs made up of single big format cells wired in series, but it is still something to think about**[41]**.

An electric circuit model is used to model the battery charging itself. As it is shown in Figure 1, the rate of the voltage charging of the EV follows the Shepherd equation **[42]**, **[43]**

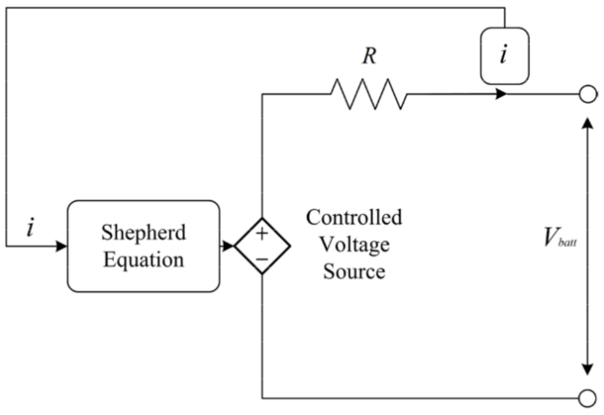


Figure 1 Electrical battery model

$$V_{batt} = E_0 - R.i - \frac{K.Q}{Q - i.t}.i^* - \frac{K.Q}{Q - i.t}.it + Ae^{-B.i.t} \qquad 2.1$$

The above formula shows the battery pack voltage (V_{batt}). E_0 is the constant battery voltage (V). R is the internal resistance (Ω). I is the battery current (A). K is the polarization constant[2] (Ω). It is an actual battery charge (Ah). Q is the battery capacitor (Ah). I* is the filtered current (A). A is the exponential zoned amplitude (V). Moreover, B is the exponential zone time constant (V).

Also, to find the SOC of the EV battery itself, the below formula can be used**[45]**:

$$SOC=100(1- \frac{it}{Q}) \qquad 2.2$$

Here, SOC is the State of Charge of the battery (%), which is the actual battery charge (Ah). Q is the battery capacity (Ah).

[2] Electric polarization refers to the separation of the center of positive charge and the center of negative charge in a material. The separation can be caused by a sufficiently high electric field[44]

Chapter 3 : Machine Learning

ML is a subfield of artificial intelligence (AI) that enables computers to automatically learn from data and previous experiences while finding patterns and making predictions with minimum human interaction. Methods of machine learning allow computers to function independently without explicit programming. ML applications are supplied with new data and can autonomously learn, evolve, and adapt. Machine learning extracts meaningful information from vast amounts of data using algorithms to find patterns and learn iteratively. Also, algorithms employ computing techniques to learn directly from data instead of depending on any preconceived model equation. During the 'learning' processes, the performance of ML algorithms improves adaptively as the number of accessible examples increases**[46]**. Deep learning, for instance, is a subfield of machine learning that teaches computers to replicate natural human characteristics, such as learning from examples. It provides superior performance parameters compared to typical ML algorithms.

Machine learning techniques are often used on a training dataset to develop a model. The trained machine learning algorithm will generate a prediction based on the established model whenever new input data is added to the algorithm. In addition, the prediction's accuracy is investigated further. Depending on its accuracy, the machine learning algorithm is used or trained again with an enhanced training dataset until it reaches the needed level.

Choosing the right machine learning algorithms is crucial to achieving accurate and reliable results. This thesis provides clear explanations for including ANN, Logistic Regression, Naive Bayes, SVM, and KNN in the research. These techniques are selected based on their unique characteristics, proven effectiveness, and suitability for the current situation.

- **ANN**: ANNs are a fascinating choice because they can handle complex patterns in data. They are like the detectives of machine learning, capable of uncovering intricate relationships and capturing nonlinearities. Since the process is complex, ANNs are well-equipped to explore and analyze the data thoroughly.

- **Logistic Regression**: Logistic Regression is a classic technique for binary classification problems. It gives interpretable results and has been widely used for decades. By including Logistic Regression, its performance can be compared against more advanced methods and gain valuable insights. Plus, it is efficient and straightforward, which is proper for simpler datasets.
- **Naive Bayes**: Naive Bayes is a simple and efficient technique, almost like a quick and clever assistant. It performs remarkably well with limited training data and is particularly useful when dealing with text classification or problems where features are independent. Despite its simplicity, Naive Bayes can handle large or sparse datasets effectively, making it a handy tool in this work.
- **SVM**: SVMs are reliable guides through a maze, helping to navigate complex classification problems. They are excellent at finding optimal decision boundaries that separate different classes, even in high-dimensional spaces. SVMs generalize well and can handle datasets with both linear and non-linear boundaries. They are particularly valuable when it has limited training examples or must balance accuracy and interpretability.
- **KNN**: KNN is a K-nearest neighbour that relies on the company of others to make decisions. It looks at the "k" closest data points to classify new instances. KNN is useful when the decision boundary is irregular or unknown and is robust to noise. It is a simple and intuitive technique that effectively handles multi-class classification problems. Adding KNN to the set of techniques ensures a well-rounded approach.

With a diverse range of machine learning techniques selected, the intention is to comprehensively investigate the situation, reveal concealed insights, and determine the optimal approaches. Each technique possesses strengths and weaknesses, enabling the consideration of complexity: interpretability, scalability, and performance. Ultimately, the aim is to provide a comprehensive analysis and choose the most suitable techniques for this research objective. Each of them is going to be explained in the next parts.

3.1 Types of Machine Learning

There are four common ways that the system can train data. Supervised, Unsupervised, Semi-Supervised, and Reinforcement Learning are the ones that are going to be explained. Due to the differences between these four methods, supervised and unsupervised learning models are often applied to distinct problems or tasks**[47]**.

3.1.1 Unsupervised learning

Unsupervised machine learning is the process of training models using unlabeled training data. It is often used to find patterns and trends in raw information and comparable group data in a specified number of groups. It is also a standard method for better understanding datasets during the early exploration phase. As the name implies, unsupervised machine learning is a more hands-off technique than supervised machine learning. A person will configure model hyperparameters like the number of cluster points, but the model will analyze massive data arrays efficiently and without human intervention. Unsupervised machine learning is, therefore, well suited to answering inquiries regarding previously undiscovered patterns and correlations within data. However, due to the reduced human oversight, using unsupervised machine learning requires particular attention to ensure accurate results. The overwhelming bulk of public data is unlabeled and unprocessed. Unsupervised learning is a powerful method for gaining insight from data by grouping data along similar qualities or analyzing datasets for underlying patterns. In contrast, since tagged data is required, supervised machine learning may be resource-heavy. Unsupervised machine learning is mainly used to**[47]**:

- Datasets are clustered based on similarities between characteristics, or data is segmented.
- Recognize the link between several data points, such as computerized music suggestions.
- Perform preliminary data analysis

There are two other forms of unsupervised machine learning:

Clustering: This refers to arranging items into clusters based on factors such as their similarities or differences. For instance, categorizing clients based on the things they buy. K-Means Clustering Algorithm, Mean-Shift Algorithm, DBSCAN Algorithm, Principal Component Analysis, and Independent Component Analysis are well-known clustering methods.

Association: Association learning identifies common relationships between the variables in an extensive dataset. It finds the relationship between different data pieces and map variables. Web traffic mining and market data analysis are two such applications. The Apriori Algorithm, the Eclat Algorithm, and the FP-Growth Algorithm are well-known algorithms that adhere to association principles.**[48]**.

3.1.2 Supervised Learning

Supervised machine learning needs labelled input and output data during the training stage of a machine learning lifecycle. A data scientist labelling the training data during the preprocessing phase is essential in training and testing a model. The model's ability to categorize new, unknown information and forecast outcomes relies on acquiring this knowledge of the link between input and output data. The term "supervised machine learning" refers to the fact that some aspect of this strategy needs human intervention. However, because of technological progress, all of these steps can be done by software. Python is one of many software that makes modelling more accessible and accurate.

The overwhelming bulk of data is raw data that has not been labelled. Human intervention is often necessary to provide reliable labels for supervised learning. Due to the necessity for extensive amounts of properly labelled training data, this may be a time-consuming and energy-consuming operation. Supervised machine learning categorises previously unseen data into known groups to anticipate future trends and changes. Constructed using supervised machine learning, a model may be taught to identify objects and the attributes that categorize them. Practising supervised machine learning methods to train predictive models is expected. Supervised machine learning models can predict new, unseen data by learning patterns between input and output data. This might be useful in predicting the direction of housing market movements or consumer spending patterns. Uses of supervised machine learning include**[47]**:

- Sorting the many forms of digital media, such as photographs, written texts, and audio recordings, into distinct categories.
- The ability to predict future events and trends by analyzing and learning from historical data.

The primary goal of supervised learning is to transfer the input variable (a) to the output variable (b) (b). Further categorizing supervised machine learning into two major categories:

Classification: Refers to algorithms that solve classification issues using categorical output variables, such as yes or no, true or false, and male or female. In the real world, spam detection and email filtering are obvious uses of this category.

Random Forest algorithms, Decision Tree Algorithms, Logistic Regression Algorithms, and Support Vector Machine Algorithms are examples of well-known classification methods.

Regression: Regression algorithms manage regression issues with a linear connection between input and output variables. It is recognized that they may forecast continuous output variables. Examples include weather forecasting and examination of market trends.

The Simple Linear Regression Algorithm, Multivariate Regression Algorithm, Decision Tree Algorithm, and Lasso Regression are popular regression methods.

The two methods of the approach used most often in machine learning methodologies are outlined. Other types of learning, such as semi-supervised and reinforcement learning, also apply to this topic. Given that these two models will not be used in this job, it does not seem that more knowledge about those topics is required. In the upcoming parts, different types of Supervised learning will be explained since, in further chapters, the model the ML creates will be trained with these methods.

3.2.1 Regression

All training data are used in the regression process to generate a single output value. This value is a probabilistic interpretation ascertained after considering the degree of correlation among the input variables. The logistic regression output comprises discrete values determined by independent variables. This method can potentially fail when confronted with nonlinear and multiple decision boundaries.

Additionally, it lacks the flexibility necessary to accurately capture the intricate relationships that exist within datasets**[49]**. There are vast numbers of regression in ML that can be found in **[49]**. However, the only type of this method that would be used in the future model of this work is Logarithmic Regression, which will be elaborated on.

3.2.1.1 Logarithmic Regression

Logistic (Logarithmic) Regression is a type of regression analysis technique utilized when the dependent variable's discrete nature is required, 0 or 1, true or false, etc. This implies that the target variable can only take on one of two possible values, and a sigmoid curve stands for the relationship between the target variable and the independent variable. In logistic regression, a logit function is essential for measuring how the target variable is related to the independent variables. You will find the logistic regression equation **[49]**.

$$\boldsymbol{f(m) = \frac{1}{1 + e^{-m}}} \qquad 3.1$$

The mentioned formula corresponds to a mathematical function known as the sigmoid or logistic function. This function finds extensive application in logistic regression for data analysis and outcome prediction. Its primary purpose is to convert any real-valued input into a value ranging from 0 to 1.

Visualized as an S-shaped curve, the sigmoid function, plays a crucial role in modelling the connection between the independent variables (the factors of interest) and the probability of the desired outcome. When referring to **'m'** in this context, a calculation that combines the independent variables with their respective coefficients is discussed. This calculation is straightforward: each independent variable is multiplied by its corresponding coefficient, and the results are summed.

$$\boldsymbol{m = \beta 0 + \beta 1 * X1 + \beta 2 * X2 + \ldots + \beta n * Xn} \qquad 3.2$$

In logistic regression, the coefficients or parameters (β0, β1, β2, ..., βn) play a vital role in capturing the relationships between the independent variables (X1, X2, ..., Xn) and the outcome. These coefficients quantify the impact of the independent variables on the probability of the outcome occurring. Applying the sigmoid function to the linear combination of these coefficients and independent variables gives a value ranging from 0 to 1, representing the estimated probability.

The sigmoid function is computed using the equation **1/(1+e^(-m))**. Here, 'm' is determined by taking the dot product of the coefficients and independent variables. The exponential term **e^(-m)** is obtained by raising the mathematical constant **'e'** (the base of the natural logarithm) to the power of the negative **'m'**. Dividing **one** by **1+e^(-m)** ensures that the output of the sigmoid function remains within the desired probability range.

The output of the sigmoid function signifies the estimated probability of the dependent variable belonging to the positive category, considering the values of the independent variables. A sigmoid function output close to 1 indicates a high likelihood of the positive category, while a value near 0 suggests a higher probability of the negative category.

During the logistic regression process, the coefficients (**β0, β1, β2, ..., βn**) are estimated by maximizing the likelihood of the observed data. This entails finding the values that closely align the predicted probabilities with the actual outcomes. Once the model is trained, the sigmoid function can generate predictions by inputting the values of the independent variables. The resulting output provides the predicted probability of the dependent variable falling into the positive category.

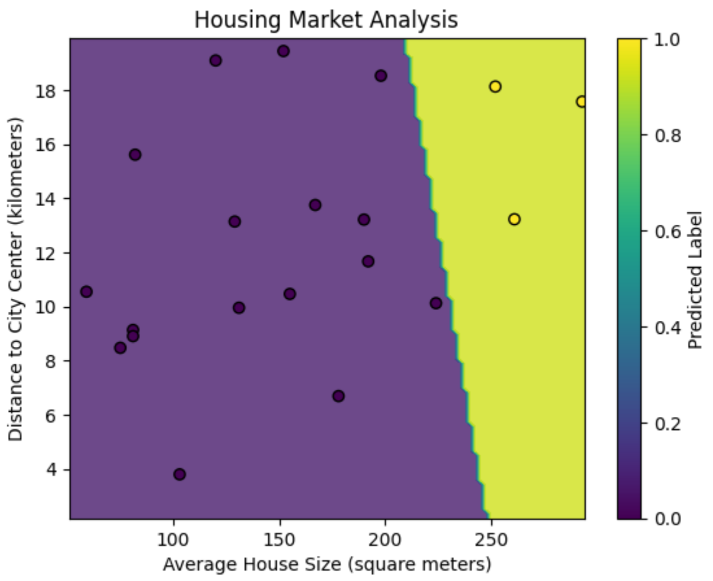


Figure 2 Logistic regression

Figure 2 Logistic Regression, the provided figure showcases a housing market analysis through a contour plot. The plot displays values for "Average House Size" and "Distance to City Center." These two features are represented on the x-axis and y-axis, respectively. The x-axis, labelled as "Average House Size," indicates the size of houses in square meters. The values along this axis correspond to various house sizes, spanning from a minimum to a maximum value. On the y-axis, labelled as "Distance to City Center," the plotted values represent the distance of each house from the city centre in kilometres. The axis covers the smallest to largest distance within the dataset. The contour lines and shaded areas on the plot represent the classification boundaries determined by a logistic regression model. These boundaries differentiate between houses that meet a certain pricing threshold and those that do not. The specific threshold is determined by the combination of "Average House Size" and "Distance to the City Center."

Points belonging to the same classification group are shaded with the same colour. These colours correspond to the colour bar on the side of the plot, which helps interpret the predicted classifications. The colour bar is labelled as "Predicted Label."

This contour plot shows how the logistic regression model segregates houses into two categories based on their average size and distance from the city centre. The figure provides an intuitive understanding of how these two features influence the classification outcome and pricing predictions.

3.2.2 Naïve Bayes

The Bayesian classification model is used for large finite datasets. It is a method of assigning class labels that employ a direct acyclic graph. The graph has one parent node and multiple child nodes. Moreover, each child node is assumed to be independent and distinct from the parent. Because the model for supervised learning in ML assists in constructing classifiers simply and straightforwardly, it works well with minimal data sets. This model is based on common data assumptions, such as independent attributes. Despite this simplification, this algorithm can be easily implemented on complex problems[49]. Bayes' theorem, mathematically represented as:

$$P(A|B) = (P(B|A) * P(A)) / P(B) \qquad 3.3$$

unlocks the door to understanding the posterior probability (P(A|B) of an event A, given evidence B. It helps to determine the probability of a particular class label (A) being assigned to the data, given the observed features or attributes (B). By calculating the likelihood of B given A (P(B|A)), the prior probability of A (P(A)), and the probability of B (P(B)), an informed estimate of the posterior probability can be arrived at.

By applying this formula within the Bayesian classification model, predictions can be made, or class labels can be assigned based on the observed features of this dataset. The model utilizes training data to estimate the probabilities Bayes' theorem requires and then applies these estimates to classify new, unseen instances.

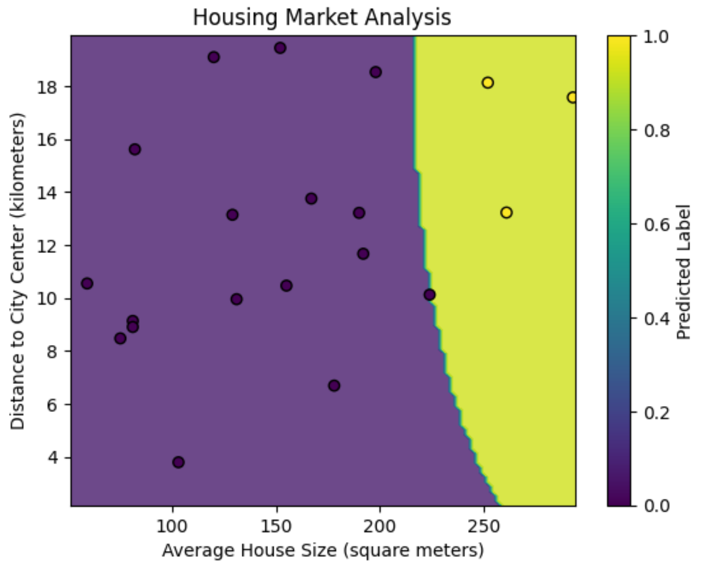


Figure 3 Naive bayes

Figure 3 the generated figure portrays a housing market analysis, visualizing the interplay between two key factors: "Average House Size" and "Distance to the City Center." The x-axis corresponds to the average size of houses, measured in square meters, providing insight into the properties' physical dimensions. On the y-axis, the distance to the city centre is depicted in kilometres, indicating the proximity of the houses to the urban core. The arrangement of contour regions on the plot delineates the model's predicted classifications of houses as high-value or not based on their features. Scatter points overlay these contours, with each point representing a specific house. The colour of each point signifies the true classification (high-value or not), allowing for a visual comparison between actual and predicted outcomes. This representation illustrates the model's ability to distinguish housing attributes, aiding in understanding how factors such as size and location influence value assessments in a simplified scenario.

Observing the contour plot facilitates the comprehension of how the Naive Bayes classifier perceives the data distribution and determines the optimal separation points between classes. As the contour lines delineate the decision boundary, one can deduce how the classifier categorizes observations into specific classes, offering a more profound understanding of the algorithm's behaviour.

Visualizing the class conditional probability distributions gives insights into how the Naive Bayes classifier leverages these distributions to make probabilistic predictions. The figure aids in understanding the foundational assumption of the Naive Bayes algorithm and provides an intuitive representation of how it separates and classifies data points based on their likelihood within each class.

3.2.3 K-nearest neighbours

K-nearest neighbours, often known as k-NN, is an algorithm for pattern recognition that locates the k people who are the closest relatives in future cases using training datasets. When doing a classification using k-NN, you will perform calculations to classify the data under the category of its nearest neighbour. If k equals 1, it will be placed in the class closest to 1. A vote by a plurality of K's neighbours decided its classification**[49]**.

$$d(x,y) = \sqrt{((x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_n - y_n)^2)} \qquad 3.4$$

The formula used to calculate the distance between two data points, typically in an Euclidean space, encompasses several steps. To begin, 'x' and 'y' represent two distinct data points within the dataset, each possessing 'n' features or attributes. The formula involves the computation of the square root of the sum of squared differences between corresponding feature values. This distance formula enables the measurement of dissimilarity between two data points, considering all the features they possess. The choice of the Euclidean distance as the metric is a common practice in this context. Several steps are generally followed when applying the k-NN algorithm to classify a new data point. Firstly, the algorithm calculates the distances between the new and existing data points in the training dataset, utilizing the aforementioned distance formula. This process establishes the proximity of the new data point to the rest of the dataset. Next, the algorithm selects the k nearest neighbours, determined by the shortest distances to the new data point. By identifying the data points closest to the new point, the algorithm gains insights into the characteristics of similar instances in the dataset.

Lastly, to assign a class to the new data point, the k-NN algorithm employs a majority vote among the classes of its k nearest neighbours. This entails determining the most frequent class among the neighbouring data points. In classification tasks, this majority vote is the basis for assigning the appropriate class label to the new data point. Following these steps, the k-NN algorithm classifies new data points by leveraging their similarities with existing data points in the training dataset. The value chosen for **'k'** plays a vital role in this process, as it determines the number of neighbours considered for classification.

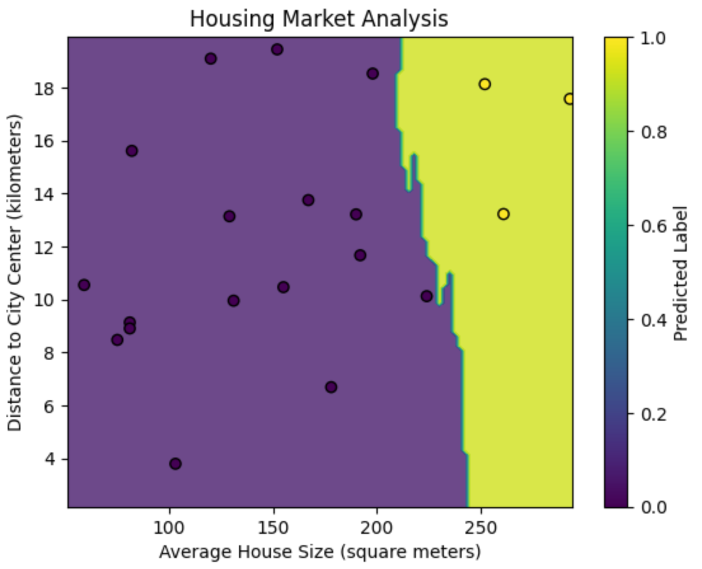


Figure 3 KNN

Figure 3 presents the KNN technique for the housing market analysis. KNN operates by grouping houses with similar characteristics together. When predicting the value of a house, KNN examines the values of its nearest neighbouring houses. If most of these neighbours are classified as high-value houses, the model predicts a high value for the house in question. This approach contrasts with Naive Bayes, as KNN focuses on the proximity of houses in the feature space rather than making probabilistic assumptions about the data. The resulting plot visually represents the outcomes of this KNN analysis, illustrating the model's predictive decisions based on the average house size and the distance to the city centre. The scatter points on the plot display individual houses from the test dataset, offering insight into how KNN classifies houses according to their surrounding neighbours, thereby simplifying the understanding of this technique's impact on housing market insights.

The curve's undulating shape results from the intricate interplay between data points and their respective class assignments. The contour lines gracefully encapsulate the classifier's understanding of the data's distribution, effectively separating regions of distinct class predictions. The choice of k and the number of nearest neighbours influences the curve's smoothness and sensitivity. Thus, the contour plot provides an intuitive visualization of the KNN algorithm's flexible adaptation to various data configurations. By examining this curve, observers gain valuable insights into how the KNN classifier leverages the spatial proximity of data points to make class predictions, thereby illuminating the algorithm's localized decision-making approach and its utility for capturing complex, non-linear relationships within data.

3.2.4 Support Vector Machines

The Support Vector Machine, sometimes known as SVM, is a method for supervised learning created in 1990. Because it can be used for either classification or regression, SVM, a method for supervised learning in machine learning, is quite popular among supervised learning models. The model's implementation works particularly well with high-dimensional spaces, although it can also be utilized productively with very modest data sets. When the algorithm is trained on a data set, SVM can also efficiently classify fresh observations. The data set is separated into two categories by SVM by creating single or multiple hyperplanes. SVM stands out from the other supervised learning models because of its unique segregation method, which gives it a performance advantage.

On the other hand, analyzing data with large dimensions can challenge this approach. The explanation is as straightforward as that SVM raises the dimensionality of the provided data set to separate it appropriately. For example, when finding solutions to linear problems, SVM will add what is known as a classifier as a feature to the feature vector. Because of this, the data set that was before only two-dimensional is now three-dimensional. Since it can distinguish between hyperplanes, support vector machines are considered discriminative classifiers. The output is created as an ideal hyperplane that classifies fresh samples. This is how the output looks. SVMs are utilized in various sectors and have close ties to the structure of the kernel.

Bioinformatics, pattern recognition, and multimedia information retrieval are a few examples of these fields. Finding a hyperplane with a maximum margin is the major objective of the Kernel function. This will assist in dividing the observations into classes based on the maximum distance between the hyperplane and the nearest point from each category. This restriction is crucial because it lowers the likelihood that the resultant hyperplane will overfit the data. SVM accomplishes the transformation from a lower-dimensional space to a higher-dimensional space by utilising a variety of kernel functions, such as similarity functions. There is a possibility that the Kernel functions will shift based on the kind of data set. Since it is not always obvious which kernel function is best suited to increase the dimensionality of the data, SVM often implements numerous kernel functions. This is done because it is not immediately evident which kernel function is best suited to increase the dimensionality of the data[49].

The formula related to SVM involves the optimization of a decision boundary, which is defined by a separating hyperplane in a high-dimensional feature space. For simplicity, a classification case with two classes is considered: positive (+1) and negative (-1). The SVM algorithm seeks to find a hyperplane that maximally separates the data points of these two classes. The formula for the linear SVM is as follows:

$$f(x) = sign(w^*x + b) \tag{3.5}$$

In this formula, 'x' represents a data point, 'w' denotes the weight vector perpendicular to the hyperplane, and 'b' is the bias term. The dot product (·) represents the sum of the element-wise multiplication of 'w' and 'x'. The 'sign' function returns +1 if the value of w*x + b is positive or zero and -1 otherwise.

During the training phase, the SVM algorithm aims to find the optimal values for **'w'** and **'b'** that define the decision boundary. The goal is to maximize the margin, the distance between the decision boundary and the closest data points from each class. These closest data points are called support vectors, which are crucial in determining the decision boundary. To find the optimal **'w'** and **'b'**, SVM employs an optimization algorithm that minimizes the following objective function:

$$\textit{minimize}\ \tfrac{1}{2}||w||^2 + C\sum \xi_i \qquad 3.6$$

$$\textit{subject to}\ y_i(w\cdot x_i + b) \geq 1 - \xi_i\ \textit{for all training examples}\ (x_i, y_i)$$

In this formula, **'||w||²'** represents the **L2** norm (Euclidean norm) of the weight vector **'w'**, and **'C'** is the regularization parameter that controls the trade-off between maximizing the margin and minimizing the training errors. The term $\sum \xi_i$ represents the sum of the slack variables (ξ_i) that allow for some misclassification or margin violations. The inequalities ensure that all data points lie on the right side of the decision boundary with a margin of at least **1 -** ξ_i.

Solving this optimization problem leads to finding the optimal **'w'** and **'b'** values, which define the decision boundary of the SVM. Once trained, the SVM can classify new data points by evaluating the w·x + b sign. If the value is positive, the data point is assigned to the positive class; otherwise, it is assigned to the negative class.

It is worth mentioning that the linear SVM is just one variant of the SVM algorithm. SVMs can also utilize non-linear kernels to map the data points into a higher-dimensional space, where a linear hyperplane can separate them. This allows SVMs to handle non-linearly separable data by effectively transforming the data into a space where linear separation is possible.

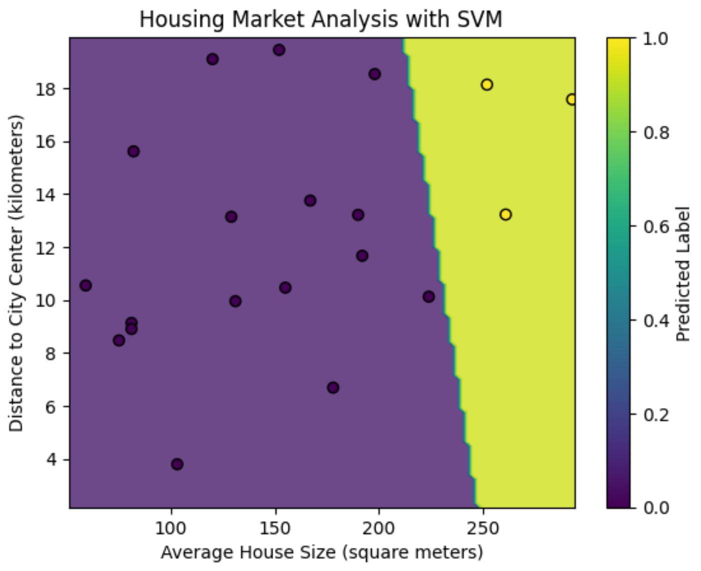


Figure 4 SVM

In this updated code, SVM has been used for the housing market analysis. SVM is a robust classification method that aims to establish a clear boundary between different classes while maximizing the separation gap between them.

The code generates synthetic housing data, representing features like "Average House Size" and "Distance to City Center." SVM is then employed to create a decision boundary that best distinguishes between high-value and non-high-value houses based on these features. Unlike previous approaches, SVM focuses on finding the optimal hyperplane that best separates the two classes in the feature space.

The resulting plot visualizes the outcomes of this SVM analysis. The x-axis signifies the average house size in square meters, while the y-axis corresponds to kilometres from the city centre. The shaded contour regions depict the model's predictions – whether houses are categorized as high-value. Scatter points overlay these contours, representing individual houses from the test dataset. This provides a tangible representation of how SVM's decision-making process categorizes houses according to their unique characteristics. This version of the code thus serves as a valuable glimpse into the impact of Support Vector Machines on the interpretation of housing market insights.

3.2.5 Artificial Neural Network

A neural network is a kind of data processing system that consists of a large number of essential processing components that are intimately associated with one another and are all interconnected with one another in a network. The organization of the cerebral cortex, the brain region responsible for higher-level thinking, serves as a model for constructing a neural network modelled after the organization of the cerebral cortex. As a direct result of this, neural networks are often capable of doing tasks that people can easily complete but that may be difficult for traditional computers to complete in some instances. As a field of study, these advancements have significantly impacted neural networks. Neural networks now present an extraordinary opportunity for research, development, and application to a wide range of issues in the real world. Neural networks, in particular, are equipped with a wide range of characteristics and capabilities not offered by any other kind of following technologies. Examples of these characteristics and abilities are reading Japanese Kanji characters and human handwriting, reading typewritten text, compensating for alignment errors in robots, interpreting very "noisy" signals (such as electrocardiograms), complex modelling that cannot be modelled mathematically, and predicting whether or not proposed loans will be successful are some of the applications of artificial intelligence[46].

In ANNs, the functions employed can exhibit versatility as they adapt to different facets of the network's operations. ANNs utilize various functions across their components, enabling them to perform intricate computations.

3.2.5.1 Activation Function

Activation functions hold a pivotal role within ANNs by introducing non-linearity to the outputs of individual nodes. They govern the firing behaviour of nodes, empowering the network to capture intricate relationships within the data. Numerous activation functions find common usage, a few of which include:

- **Sigmoid Function**: Defined as **f(x) = 1 / (1 + exp(-x))**, the sigmoid function maps input values to a range between 0 and 1. It exhibits a smooth curve suitable for binary classification problems.

$$f(x) = 1 / (1 + exp(-x)) \qquad 3.7$$

- **Rectified Linear Unit (ReLU)**: Captured by $\boldsymbol{f(x) = max(0, x)}$, ReLU functions assign a value of 0 to negative inputs and maintain positive inputs unchanged. ReLUs are known for their simplicity and effectiveness in deep learning architectures.

$$\boldsymbol{f(x) = max(0, x)} \qquad 3.8$$

- **Hyperbolic Tangent (Tanh)**: Mathematically expressed as $\boldsymbol{f(x) = (exp(x) - exp(-x)) / (exp(x) + exp(-x))}$, the hyperbolic tangent function exhibits an S-shaped curve, mapping input values to a range between -1 and 1. It is often employed in networks aiming for outputs within this range.

$$\boldsymbol{f(x) = (exp(x) - exp(-x)) / (exp(x) + exp(-x))} \qquad 3.9$$

3.2.5.2 Loss Function

Loss functions gauge the disparity between the predicted output of the network and the actual output. The choice of a particular loss function hinges on the specific task being addressed. Some commonly utilized loss functions encompass:

- **MSE**: MSE computes the average squared difference between predicted and actual values. It is widely used for regression problems.
- **Cross-Entropy Loss**: Employed for classification tasks, cross-entropy loss measures the dissimilarity between predicted class probabilities and the true class labels.
- **Binary Cross-Entropy Loss**: Similar to cross-entropy loss, binary cross-entropy loss is utilized when dealing with binary classification problems.
- **Categorical Cross-Entropy Loss**: Categorical cross-entropy loss extends the concept to multi-class classification tasks, evaluating the deviation between predicted probabilities and true class labels.

Choosing an appropriate loss function when optimizing a machine-learning model for the task was influenced by the thoughtful consideration of the current problem's inherent characteristics and the model's desired outcomes. In the context of this work, the decision to employ the MSE as the selected loss function was made.

MSE, a well-established metric frequently utilized in regression tasks, resonated with these objectives for several compelling reasons. Firstly, MSE possesses an intuitive and easily interpretable quality, as it quantifies the average squared discrepancies between predicted values and the true target values. This aspect aids in comprehending the magnitude of errors made by the model, making it conducive for transparent evaluation.

Furthermore, problem context inherently necessitates a keen focus on the magnitude of errors, which MSE prioritizes. Given that the primary aim is to minimize the deviations between predicted and actual values, MSE aligns seamlessly with the objective. By minimizing the squared errors, MSE punishes large discrepancies more significantly and fosters an emphasis on accurate predictions while accommodating outliers that could otherwise disproportionately impact other loss functions.

Additionally, from a computational perspective, MSE provides smooth gradients that facilitate efficient optimization techniques. The continuity of the loss landscape supported by MSE aids in the convergence of optimization algorithms, contributing to more stable and rapid model training.

3.2.5.3 Optimization Algorithms

Optimization algorithms drive the adjustment of weights and biases during the training phase of the network. They dictate the precise formulas to modify these parameters based on the error gradients. Unique optimization algorithms include:

- **Gradient Descent**: A fundamental optimization algorithm, gradient descent iteratively updates the parameters toward the steepest descent to minimize the error.
- **Stochastic Gradient Descent (SGD)**: An extension of gradient descent, SGD computes parameter updates based on small random subsets of the training data, enhancing efficiency.
- **Adam**: An adaptive optimization algorithm, Adam combines momentum-based methods and root-mean-square propagation concepts to achieve efficient convergence in various scenarios.
- **RMSprop**: RMSprop adjusts the learning rate adaptively based on the magnitude of recent gradients, facilitating faster convergence and improved stability.

Consequently, while no single function encapsulates all aspects of ANNs, a diverse range of functions, such as activation functions, loss functions, and optimization algorithms, play integral roles within the network. The selection and application of these functions rely upon the unique requirements and characteristics of the neural network and the specific task at hand.

Both learning and recall are essential to the functioning of any artificial neural network. The connection weights are modified throughout the learning process in response to information from the input buffer. When a learning example is presented to the input buffers, the network "learns," or adapts, according to a rule that specifies how the connection weights should change. The network's ability to recall its learned responses to new inputs is called "recall."**[46]**.

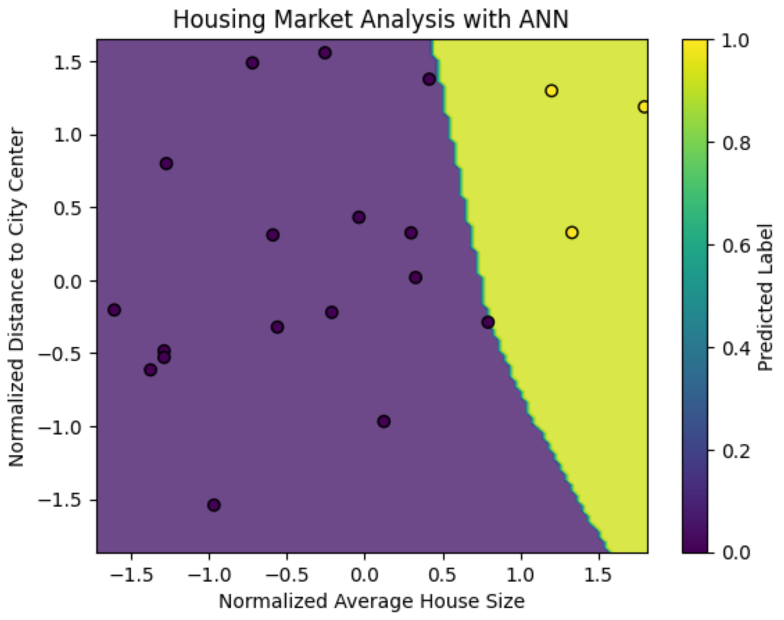


Figure 5 ANN

In Figure 5, ANN has been used for the household marketing prices. The code commences with the generation of synthetic housing data, encompassing two pivotal features: "Average House Size" and "Distance to City Center." These features indicate each house's physical attributes and geographical location.

Before proceeding with the analysis, an essential procedure of feature normalization is executed. This normalization process is facilitated by using the `StandardScaler`, which standardises the values of the features. This standardization is paramount to ensure equitable treatment of the features during the subsequent training of the model, thereby mitigating the possibility of disproportionate influence by any single feature.

Following the normalization, the data division into training and testing sets ensues, adhering to established practices within machine learning. The construction of an Artificial Neural Network follows suit, featuring a singular hidden layer housing ten neurons and employing the ReLU activation function. Subsequently, the network's training ensues using the normalized training data, allowing it to discern intricate relationships between the features of the houses and their corresponding values.

Upon the culmination of the training process, the model's predictive capabilities are brought to the forefront. Predictions are performed on the testing dataset, and the subsequent outcomes are visually elucidated through a contour plot. This graphical representation offers a two-dimensional portrayal, with the normalized "Average House Size" on the x-axis and the normalized "Distance to City Center" occupying the y-axis. The contour regions effectively encapsulate the model's predicted classifications, denoting whether a given house attains high-value status based on the attributes normalized for analysis.

The overlaying of scatter points upon these contour regions offers a granular perspective, with each point signifying an individual house from the test dataset. Each point's colour indicates its factual classification (high-value or not), with the deliberate inclusion of black edges around the points to provide a clear demarcation against the backdrop of the contour regions. This visual representation is a tangible manifestation of the model's adeptness in categorizing houses predicated upon their normalized features. By affecting the normalization of features, the model ensures impartial and unbiased predictions, thereby forestalling the undue sway of any solitary feature during the analysis. This version of the code is an illustrative testament to the strategic deployment of Artificial Neural Networks for extracting insights from housing market data.

3.3 Classification Evaluation

Classification methods are often evaluated based on accuracy, although this metric may be misleading when imbalanced classes are used. Metrics of recall, precision, and F-measurement are used to investigate the effectiveness of classification strategies in solving different related problems [50]. Below are formulas for the said criteria for assessment.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \qquad 3.10$$

$$Recall(R) = \frac{TP}{TP+FN} \qquad 3.11$$

$$F\beta = (1 + \beta 2).\frac{Precision.Recall}{\beta^2 Precision+Recall} \qquad 3.12$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Yi- \hat{yi})^2 \qquad 3.13$$

$$Precision = \frac{TP}{TP+FP} \qquad 3.14$$

True Positives (TP) are several instances predicted as abnormal. False Positives (FP) are several instances predicted as abnormal but usual. True Negative (TN) is a number of instances predicted as expected and normal. False Negative (FN) is several instances predicted as normal but abnormal[50]. Also, for the mean square error or MSE, as explained before, **Y** represents the actual number, and **ŷ** represents the output number of the model itself. Beta is a parameter that determines the recall weight in the F-beta score. It allows you to control the trade-off between precision and recall. If **$\beta > 1$**, the recall has more importance, and if **$\beta < 1$**, precision is more important. When **$\beta = 1$**, the F-beta score is the harmonic mean of precision and recall.

The rationale behind the prioritization of Accuracy, F1 Score, and MSE in the context of the machine learning study was informed by discerning factors that ensured a judicious selection of evaluation metrics. These factors encompassed considerations related to problem relevance, methodological consistency, communication efficacy, interpretative clarity, and resource optimization.

Relevance to the Problem: The metrics in focus—Accuracy, F1 Score, and MSE—were chosen due to their inherent alignment with the quintessence of the problem domain. While Accuracy underscored the holistic assessment of correct predictions, F1 Score's emphasis on precision and recall equilibrium and MSE's quantification of prediction deviation collectively converged to harmonize with the primary objectives spanning both classification and regression accuracy.

Consistency for Comparative Analysis: The deliberate preference for a uniform set of metrics—Accuracy, F1 Score, and MSE—was underpinned by the pursuit of equitable comparative analysis across diverse machine learning methodologies, namely ANN, SVM, KNN, Logistic Regression, and Naive Bayes. This uniformity aimed to engender an equitable footing for assessing the performance of each technique, fortifying the analytical robustness while facilitating dispassionate scrutiny of their merits and demerits.

Facilitation of Effective Communication: The selection of the metrics above—Accuracy, F1 Score, and MSE—was rooted in their widespread comprehension and interpretability. The deliberate choice of these metrics streamlined the communication of findings to a varied audience spectrum, engendering a lucid portrayal of the proficiency of the deployed techniques and the ensuing implicational ramifications.

Equilibrium between Complexity and Clarity: While acknowledging the pertinence of specificity as an evaluative gauge, the focal point of the study was to maintain an equilibrium between metrics' elucidative potential and complexity management. The emphasis lay in channelling attention towards a parsimonious selection of metrics that authentically resonated with the problem's purview and enjoyed substantial acknowledgement within the research community. This approach sought to ensure accessibility and pragmatic utility of the analytical outcomes.

3.4 Feature Selection

In developing predictive models, feature selection is critical in minimizing the number of input variables used. By reducing the number of input variables, computational costs can be lowered, and in certain cases, the model's performance can be improved. Feature selection approaches based on statistical measures analyze the relationship between each input variable and the target variable, selecting the variables with the strongest association **[51]**. The choice of statistical measures depends on the data types of the input and output variables, but these approaches effectively identify relevant features. However, practitioners may face challenges in selecting the appropriate statistical measure for a given dataset while implementing filter-based feature selection.

The main objective of feature selection is to eliminate irrelevant or redundant predictors from the model. Many models, particularly those based on regression slopes and intercepts, estimate parameters for each term included in the model. Consequently, non-informative variables can increase prediction uncertainty and reduce overall model efficiency. It is important to note that feature selection can be performed using supervised or unsupervised techniques. Unsupervised feature selection disregards the outcome when removing features **[51]**.

In this study, the evaluation of the most useful features will be conducted using the Correlation Matrix **[52]**and manual accuracy comparison. This approach identifies features appropriate for the system and presents them in the feature selection subset **[51]**. The correlation matrix provides information about the correlation between features and the target variable, while manual accuracy comparison helps identify the combination of features that yields the best model performance.

3.4.1 Features Correlation Matrix

A correlation matrix plays a significant role in analyzing data by revealing the connections between multiple variables. It provides a convenient tabular representation of correlation coefficients, capturing the relationships between each variable and all others in the dataset [53]. This powerful statistical tool finds applications in diverse research domains, including data analysis, statistical modelling, and scientific experimentation. It aids in identifying patterns and associations within the dataset and serves as input for advanced techniques such as principal component analysis, factor analysis, and multiple regression.

Each cell represents the correlation coefficient between a pair of variables within the correlation matrix. These coefficients, ranging from -1 to 1, reveal the strength and direction of the relationships. A coefficient of 1 indicates a perfect positive correlation, -1 indicates a perfect negative correlation, and 0 suggests no correlation. The proximity of the coefficient to -1 or 1 indicates the strength of the relationship. Positive correlation implies that an increase in one variable corresponds to an increase in another, while negative correlation suggests an inverse relationship.

Moreover, the correlation matrix unveils highly correlated features, known as multicollinearity. Identifying and managing such features is crucial as they may not contribute unique information to the model. In multicollinearity cases, retaining the feature that exhibits a stronger correlation with the target variable is advisable. It is important to emphasize that correlation matrices only reveal linear relationships between variables and should not be misconstrued as indicating causation

3.4.2 Features Impression Rating

Feature importance analysis is a valuable technique that assesses each feature's significance in influencing a model's performance. By comparing the model's predictions with and without a particular feature, can be understood its impact on the overall accuracy [54]. The exploration of "Feature Importance Rating" within this research delves into an essential aspect of machine learning aimed at discerning the contribution of individual features in predicting the target variable. This analysis assumes significance in comprehending the relative significance of different features towards the model's predictive power. By identifying which features hold greater influence, one can streamline feature selection processes, enhance model interpretability, and potentially mitigate issues stemming from irrelevant or redundant attributes.

The ANOVA F-test (Analysis of Variance F-test) is vital. Its passive utilization lies in assessing the extent of variance among the means of distinct feature categories and subsequently deducing whether the differences observed are statistically significant. In essence, the ANOVA F-test quantifies the variability attributed to the categories relative to the variability within each category. Its application discerns whether the means of different categories are genuinely distinct or if the differences arise from random fluctuations. By employing the ANOVA F-test, this research methodically gauges the influence of individual features on the target variable while upholding a rigorous statistical framework.

$$\boldsymbol{F = \frac{Between - group\ variability}{Within - group\ variablity}} \qquad 3.\mathbf{15}$$

- The Between-group variability signifies the degree to which the means of different feature categories deviate from one another. It captures the extent of variation between the group means and provides insights into whether certain feature categories significantly differ from others.
- The Within-group variability represents the dispersion within each feature category. It measures how individual observations within a group deviate from the group's mean. This measure is pivotal in understanding the overall variability present within each category.

The F statistic, derived from this formula, serves as an evaluative metric. A higher F value suggests that the differences between group means are substantial compared to the inherent variability within each category. This scenario points towards statistically meaningful differences among feature categories, implying the presence of influential features.

Conversely, a lower F value indicates that the differences between group means are more aligned with random variations within each category. In such cases, the feature categories may not exert as significant an influence on the target variable.

3.5 Correlated Time

Correlation time refers to the duration or time scale over which a correlation between two variables or phenomena persists. It quantifies the temporal relationship and provides insight into how long the influence of one variable can affect another**[55]**. In the initial phase of this study, the assumption was made that there is no correlation time between features. This means the default assumption was that each minute is independent of the others. However, to further analyze the impact of correlation on feature selection and model performance, different correlation times (short, medium, and long) were introduced using lagged steps.

By incorporating various correlation times, the assessment can be made of how noise correlations influence feature selection and the effectiveness of machine learning models. Short correlation times indicate a weak correlation between consecutive time steps, while medium and long correlation times represent a stronger correlation. Failure to account for correlated noise may result in selecting irrelevant or unimportant features for the model, potentially leading to overfitting or underfitting. Therefore, considering different correlation times and evaluating model performance, this can be identified as the most suitable set of features to optimize the model's performance.

3.5.1 Lagged Steps

Using lagged steps in data analysis is crucial in capturing the interdependencies between current and previous data points. By incorporating lagged steps of varying sizes, correlations over different time scales can be effectively examined **[56]**.

For this research, a dataset comprising more than 100,000 data points spanning over 120 days was gathered to predict values hourly. Lagged steps are applied during the data processing to account for various correlation times. Specifically, three distinct lagged step sizes are considered: 15, 30, and 60, corresponding to time intervals of 15 minutes, 30 minutes, and 1 hour, respectively. These lagged steps enable the capture of short, medium, and long correlations.

By integrating these various correlation times, this study investigates the impact of incorporating lagged steps in the feature selection process and assesses the performance of the machine learning techniques across different time horizons. The lagged correlation formula calculates correlations using lagged steps in time series analysis. This formula quantifies the relationship or similarity between two variables at different time points. It is defined as follows:

Correlation coef (lagged) = **3.16**

Cov(X_t, Y_{t-l}) / (std(X_t) * std(Y_{t-l}))

Covariance(**X_t**, **Y_{t − l}**) represents the covariance between variable X at time **t** and variable Y at a lagged time point **t-l**. **Std**(**X_t**) and **std**(**Y_{t − l}**) denote the standard deviations of variable X at time **t** and variable Y at the lagged time point t-l, respectively.

The lagged time point **(t-l)** represents the time shift or delay between the analysed variables. The correlation coefficient is obtained by calculating the covariance between the variables at the specific lagged time point and dividing it by the product of their standard deviations. This coefficient indicates the strength and direction of the linear relationship between the variables at the given lag. A positive correlation coefficient signifies a positive relationship, while a negative correlation coefficient indicates an inverse relationship. A correlation coefficient close to zero suggests a weak or negligible linear relationship.

3.6 Correlated Noise

Correlated noise is a phenomenon that affects the performance of many different kinds of systems. The output of a system is subject to noise if it experiences any random fluctuation or disruption. In contrast to truly random noise, correlated noise displays some degree of correlation or pattern. Electronic circuits are a prominent source of correlated noise because voltage and current changes may cause undesired variations in the output signal. Temperature, humidity, and electromagnetic interference are all potential causes of these shifts. Some correlations may be deterministic and amenable to mathematical modelling. Biological and social systems are only two examples of systems where correlated noise may emerge. For instance, behaviour correlations may emerge in a group of animals due to individuals' movement patterns being impacted by those of other animals. Signal processing, communication systems, and control theory are just a few disciplines that benefit from understanding correlated noise. Learning how noise affects a system's performance and coming up with solutions to that problem is a common need in these areas. A covariance matrix may explain correlated noise by describing the statistical correlations between the various noise sources. This matrix's help may predict and improve The system's performance.

Correlated noise may improve a model for various reasons. First, it may replicate real-world noise that has structure or correlation. In certain physical systems, several sources may cause correlated noise. Correlated noise improves system representation and prediction. Second, correlated noise tests model resilience. A model that performs well under ideal circumstances but fails when subjected to correlated noise may not be robust enough to manage real-world noise. Thirdly, correlated noise improves model generalization. The model will perform well on unknown data if it can learn to extract meaningful features despite associated noise**[57]**.

A kind of correlated noise known as Gaussian correlated noise is characterized by its use of a Gaussian distribution. It is often used in real-world modelling systems, particularly those in which the noise is not entirely random but exhibits some degree of connection or pattern. Utilizing a multivariate Gaussian distribution is a typical approach that may be used to generate Gaussian correlated noise. In this technique, the noise is produced by selecting values at random from a multivariate Gaussian distribution that has already had its covariance matrix specified. It is possible to represent the correlations between the various noise sources using the covariance matrix, which describes their statistical connections **[58]**.

$$n(t) = Lx(t) + z(t) * \sigma * \sqrt{(1 - L^2)} \tag{3.17}$$

In the given equation, **n(t)** represents the noise sample at a particular time. L denotes the Cholesky factorization matrix, while **x(t)** is the original data sample at that specific time. Furthermore, **z(t)** signifies the uncorrelated noise sample, and σ represents the standard deviation of the noise.

The intensity of the noise can be modified by its influence by multiplying the uncorrelated noise (**z(t)**) with the desired standard deviation (**σ**). Increasing the value of σ generates noise with a wider range of values, resulting in a higher standard deviation and a more pronounced impact on the data and conversely, reducing σ yields noise with a narrower range, leading to a lower standard deviation and a less noticeable effect on the data.

By carefully adjusting the scaling factor, the standard deviation of the noise can be customized to suit specific requirements. This flexibility enables the generate of noisy data with varying degrees of variability and dispersion tailored to this work's unique needs. In this research, six exceptional levels of standard deviation are chosen for this work: 0.01, 0.1, 1, 2, 4, and 6. By subjecting different machine learning techniques to these diverse noise levels, valuable insights into the resilience and adaptability of these methods when confronted with varying magnitudes of noise are gained.

Chapter 4 : Predicting EV State based on Data from Gridlab-d

Gridlab-d, an advanced software tool developed by the U.S. Department of Energy's National Renewable Energy Laboratory (NREL), plays a pivotal role in forecasting the behaviour of EVs by utilizing accurate data. This powerful simulation platform empowers researchers and engineers to model and analyze power distribution systems, specifically focusing on integrating EVs. By leveraging real data and employing sophisticated algorithms, Gridlab-d enables precise predictions of EV behaviour, leading to optimized EV charging strategies and efficient power grid management. Gridlab-d is an indispensable tool for researchers and policymakers seeking a deeper understanding of the intricate interactions between EVs and the power grid. By utilizing real data, including power consumption patterns, and incorporating additional factors such as time of day and battery SOC, Gridlab-d achieves high accuracy in predicting EV states. This predictive modelling capability equips grid operators with the foresight to anticipate EV charging demand, optimize grid resources, and ensure grid stability**[59]**.

One of the strengths of Gridlab-d lies in its ability to integrate real data into simulations. By incorporating reliable power consumption data from trusted sources like NREL or other reputable data providers, Gridlab-d offers a realistic representation of EV behaviour and its impact on the power grid. This data-driven approach enhances the reliability and practicality of the predictions generated by Gridlab-d.

Gridlab-d empowers researchers to explore diverse scenarios and assess the influence of various factors on EV states. By considering parameters such as time of day, battery SOC, and charging patterns, Gridlab-d can accurately simulate EVs' charging and discharging behaviour. This level of detail and customization allows for a comprehensive analysis of EV integration and identifying optimal charging strategies.

Furthermore, Gridlab-d facilitates the evaluation of emerging technologies and grid management strategies concerning EV integration. Researchers can assess the impact of different charging infrastructures, renewable energy sources, and demand response programs on EV behaviour and grid stability. This capability contributes to ongoing efforts to develop sustainable transportation systems and build a more resilient and efficient power grid.

4.1 Utilizing Gridlab

In the mentioned thesis, the powerful Gridlab-d software was utilized for the research, successfully implementing it to generate insightful data. To ensure a comprehensive analysis, a combination of preexisting templates provided by Gridlab-d was employed, customized according to the specific requirements of the study. Additionally, incorporating input data, consisting of grid power consumption information, into the simulations further enhanced the research.

By leveraging the capabilities of Gridlab-d, the interactions between EVs and the power grid were accurately modelled and analyzed. The user-friendly interface and extensive software documentation facilitated seamless and confident navigation. Realistic charging and discharging behaviours of EVs were simulated through meticulous parameter selection, such as considering the time of day and battery SOC. Some of the papers and publications that have been published their work based on this software can be mentioned here **[3], [60]–[62]**

By utilizing Gridlab-d, the prediction of EV states with high accuracy was achieved. The integration of real data, including grid power consumption patterns, played a significant role in enhancing the reliability of the obtained results. This approach generated meaningful insights into EV behaviour, charging strategies, and their impact on grid performance.

The thorough understanding and utilization of Gridlab-d in the thesis exemplify the proficiency in using this software for research purposes. By following established methodologies, incorporating custom input data, and interpreting the results within the study's context, the ability to leverage Gridlab-d as a valuable tool for analyzing the integration of EVs into the power grid was showcased.

4.2 Predesigned Scenario for EVs and Load Implementation

In the predefined scenario, there are four EVs with different SOCs ranging from 30% to 90%. The grid power consumption of the loads is also available as input data, and EVs can be in one of three states: Idle, G2V, and V2G. The user can choose the desired state for their car, represented by the values 1 for V2G/G2V/Idle, -1 for only charging, and 0 for the disconnected car.

In the proposed scenarios, which consider the grid power consumption ranges and the periods of 3 p.m. to 12 a.m. and 12 a.m. to 6 a.m., the activation of EVs can be strategically determined based on their SOCs, grid power consumption, and time of the day. These scenarios aim to optimize the utilization of EVs while considering grid conditions and user preferences.

- **Scenario 1**: Grid Power Consumption Less Than 3 kW During the period of 3 pm to 12 am, EVs with lower SOC (e.g., 30% to 60%) can remain idle or disconnected, conserving energy during the period of low power consumption. EVs with higher SOC (e.g., 70% to 90%) can be activated in G2V mode, discharging surplus energy back to the grid. During the 12 am to 6 am, all EVs, regardless of SOC, can be activated in G2V mode to take advantage of the lower power demand and facilitate charging.
- **Scenario 2**: Grid Power Consumption Between 3 kW and 9 kW During the time span of 3 pm to 12 am, EVs with lower SOC (e.g., 30% to 60%) can remain idle or disconnected, optimizing power consumption. EVs with higher SOC (e.g., 70% to 90%) can contribute to the grid by entering G2V mode. During the time span of 12 am to 6 am, all EVs, regardless of SOC, can be activated in G2V mode to charge their batteries efficiently during the night when power demand is low.

- **Scenario 3**: Grid Power Consumption More Than 9 kW During the time span of 3 pm to 12 am, EVs with lower SOC (e.g., 30% to 60%) can remain idle or disconnected, conserving energy due to high power consumption. EVs with higher SOC (e.g., 70% to 90%) can be activated in Vehicle-to-Grid (V2G) mode to supply energy back to the grid, supporting load management. During the time span of 12 a.m. to 6 a.m., all EVs, regardless of SOC, can be activated in G2V mode to charge their batteries, taking advantage of the lower power demand during the night.

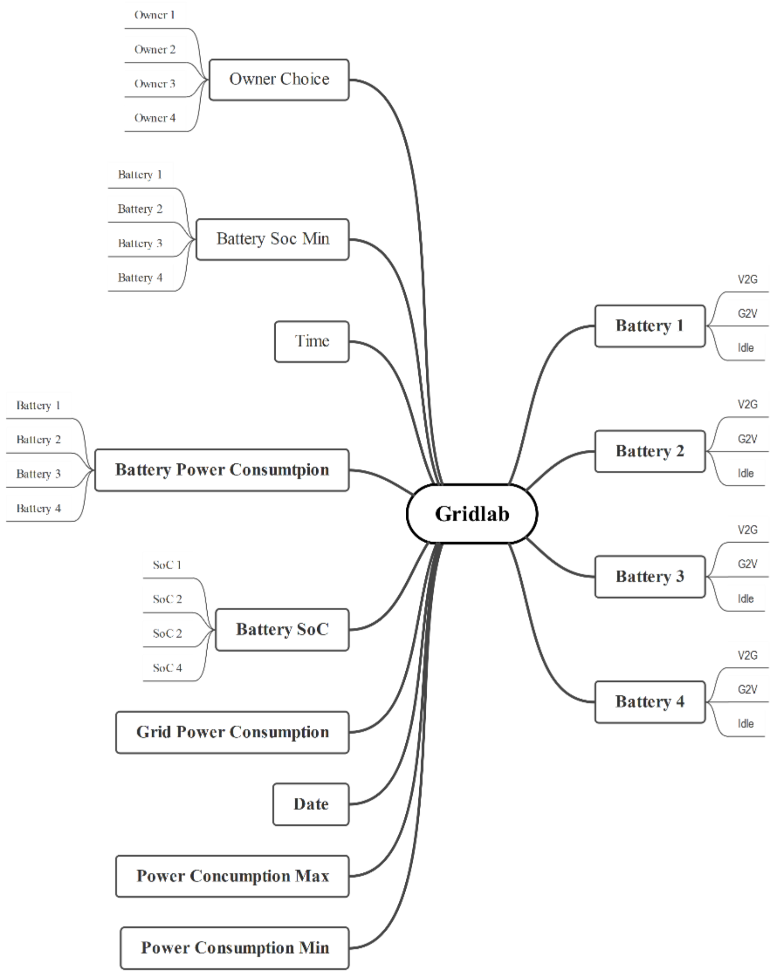


Figure 6 Gridlab input/output

Figure 6 shows the process of inputs and outputs in the Gridlab itself. In the context of this research, the decision to focus on the time span from 3:00 PM to 12:00 AM and from 12:00 AM to 6:00 AM for EV battery state prediction is underpinned by a combination of well-founded considerations. These time intervals have been chosen based on the assumption that users are typically not at home during these hours, a premise that lends itself to several beneficial aspects of the research methodology:

- **Stable Charging Patterns:** By concentrating on time frames when users are less likely to interact with their electric vehicles (EVs), the research capitalizes on the stability of charging routines. This focused approach inherently leads to a more consistent and reliable dataset. The patterns established during these hours can better represent typical EV battery charging and discharging behaviours, enhancing the accuracy of predictions and reducing the influence of sporadic charging activities.
- **Reduced Noise from Human Activities:** The selected periods align with periods when domestic activities and mobility are relatively subdued. This alignment minimizes the potential interference from household activities, such as appliance usage, which could introduce noise into the dataset. Consequently, the model's capacity to discern the distinct impact of EV battery dynamics on the grid is improved.
- **Peak Demand Mitigation:** The omission of peak usage hours from the analysis allows the research to focus on time intervals characterized by diminished grid demand. This strategic choice ensures that the model hones in on instances when the contributions of EV batteries to grid stability and power supply are most pronounced, thereby optimizing the impact of their integration.
- **Simplified Model Interpretation:** Concentrating the analysis within specific time windows simplifies the interpretation of the model's outcomes. This simplicity translates to a more intuitive and actionable model that can be readily understood and embraced by stakeholders within the energy sector.
- **Efficient Resource Allocation:** The temporal boundaries are selected to align with periods of reduced user activity, which leads to the efficient allocation of computational resources and

model training efforts. This streamlined focus enhances the utility of available resources and ensures that efforts are concentrated where they are most impactful.

- **Alignment with Grid Load Dynamics:** The chosen time intervals correspond to moments when grid load typically experiences distinct shifts, transitioning from periods of peak demand to off-peak conditions. This alignment empowers the model to effectively capture the pertinent load patterns and fluctuations, resulting in predictions that accurately reflect the dynamic nature of the grid.
- **Reduced User Interaction Variability:** The underlying assumption of limited user engagement during the chosen time frames contributes to reduced variability arising from EV interaction behaviours. This homogenization of data assists in maintaining the consistency of the dataset and minimizing the potential influence of outlier effects on the predictive model.

In the context of my research, power consumption data spanning the years 2006 to 2010 was acquired from the Kaggle platform[63]. Although the dataset does not explicitly specify the associated city, the legitimacy of this data can be substantiated through the following considerations, intended to assure my supervisor of its credibility:

- **Reliability of Data Source:**
    - Kaggle is recognized as a reputable and well-established platform known for curating datasets of high quality. The availability of the dataset on this platform suggests that certain quality control measures and assessments have been applied before its distribution.
- **Consistency and Comprehensive Coverage:**
    - The dataset consistently encompasses the timeframe from 2006 to 2010, contributing to its reliability. The continuity of data over this duration enhances its value for conducting longitudinal analyses and identifying trends.
- **Attributes and Granularity of Data:**

    - The structural attributes of the dataset, as well as its detailed information on power consumption patterns and chronological timestamps, substantiate its legitimacy for investigating power usage behaviours.
- **Characteristics of Data Distribution:**
    - Analyzing the distribution patterns of power consumption values across time can provide insights into the data's authenticity. The dataset's credibility is reinforced if these patterns align with expected household energy usage trends.
- **Corroboration with Existing Scholarship:**
    - The dataset can be compared with prior literature or similar datasets that explore power consumption trends within a comparable timeframe. Demonstrating that the dataset's patterns align with established trends enhances its validity.
- **Transparency in Methodology:**
    - Transparently outlining the preprocessing and validation steps employed before integrating the dataset into the research underscores a systematic approach. Detailing procedures such as data cleaning, outlier detection, and normalization enhance the dataset's reliability.
- **Sensitivity Analysis and Robustness Checks:**
    - Executing sensitivity analyses or robustness checks using the dataset can provide insights into its resilience against variations. This approach substantiates the dataset's utility for addressing research inquiries, even without specific location details.
- **Acknowledgment of Limitations:**
    - While underscoring the dataset's strengths, it is equally important to acknowledge its constraints. Mentioning the lack of explicit city information and its potential ramifications fosters an open and transparent discussion.

By amalgamating and presenting these points within the thesis, a compelling case can be established for the legitimacy of the power consumption dataset procured from Kaggle. Such an approach fortifies the dataset's appropriateness for integration into the research despite the absence of specific city attribution.

The origin of the information needs to be brought up first in the given explanation. However, in the amended model, the source is expected to be reduced in step size, and the phase-to-phase voltage rms of the source is 208 V. This is because the original model was flawed. In addition, the frequency of 60 Hz is being used for consideration since it is standard in North America. Also, it is assumed that there would not be a significant amount of power loss utilizing this model; the assumption is that the source's internal inductance and resistance will be relatively low. Similarly, for the hypothetical fleet, it is assumed that the considered fleet consists of four EVs and four houses as loads. These data sources are being utilized in this developed model[63].

About the EV fleet part, it is assumed that in this grid, there are only four houses as the load, and their power consumption comes from the open source data[63]**.** Also, four EV cars with different amounts of Soc, 30% to 90%, make the grid.

4.3 Implementing the ML Model

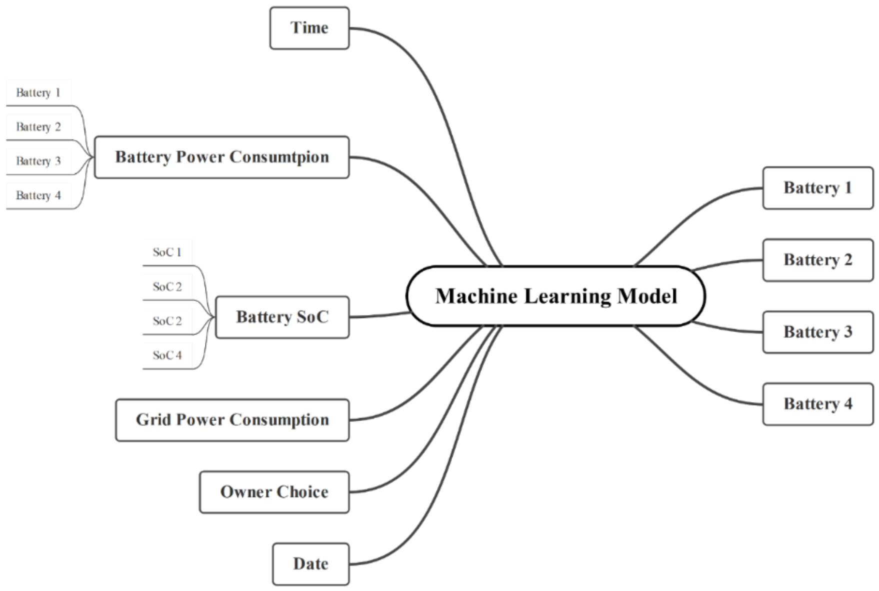


Figure 7 Ml network for one-hour ahead prediction

After carefully collecting all the necessary data, it becomes crucial to categorize them into appropriate groups of inputs and outputs for the ML model. In the context of this research, six distinct inputs are identified as crucial for the model's performance. These inputs include:

- **Time of the day**: The specific time the data is recorded, enabling the model to capture temporal patterns and variations.
- **Battery power consumption**: The amount of power consumed by the battery, providing insights into the energy usage patterns of electric vehicles (EVs).
- **Battery SOC**: The level of charge remaining in the battery at a particular point in time, indicating the available energy for the EVs.
- **Grid power consumption**: The overall power consumption of the grid which helps analyze the demand and load fluctuations.
- **Owner choice**: The user's preference regarding their EV's charging behaviour, allowing for customization and evaluation of different charging strategies.

- **Date of the day**: The specific date when the data is recorded, enabling analysis of seasonal and calendar-based variations in EV behaviour.

Alongside these inputs, each battery's state will be considered an output. The battery state reflects the current condition and performance, encompassing factors such as its health, efficiency, and remaining lifespan.

In selecting these six distinct inputs for the machine learning model, The relevance to the research objectives and the potential impact on the model's performance were considered. These inputs were chosen based on their ability to capture essential aspects of the electric vehicle (EV) charging behaviour and its interaction with the grid. Each input serves a specific purpose in enriching the model's understanding and predictive capabilities:

- **Time of the day:** This input captures temporal patterns and variations in EV charging behaviour, allowing the model to discern charging trends during different hours. This is crucial for understanding peak demand periods and optimizing charging strategies accordingly.
- **Battery power consumption:** Monitoring battery power consumption provides insights into the energy utilization patterns of EVs. This information can help identify efficient charging practices, reduce energy costs, and promote sustainability.
- **SOC:** The battery SOC input informs the model about the available energy within the EV's battery. It is vital for predicting charging requirements ensuring the vehicle's energy needs are met while avoiding overcharging or unnecessary discharges.
- **Grid power consumption:** This input reflects the broader energy demand on the grid. Analyzing grid power consumption helps the model recognize periods of high demand and adapt EV charging patterns to alleviate strain on the grid during peak usage.

- **Owner choice:** Incorporating owner preferences regarding charging behaviour introduces a customizable dimension to the model. By determining user choices, the model can provide tailored recommendations that align with individual user needs and priorities.
- **Date of the day:** Including the date allows the model to account for seasonal and calendar-based variations in EV behaviour. This is essential for understanding long-term trends and adapting charging strategies to accommodate changing seasons or holidays.

These six inputs were selected because of the need to balance data richness and model complexity. Each input contributes unique information that enhances the model's ability to predict and optimize EV charging behaviour in response to grid dynamics. Focusing on these specific inputs, the model comprehensively understands the interplay between EVs, energy consumption, and grid demands. This approach ensures the model's effectiveness and aligns with the research's scope and objectives, as outlined in my thesis.

To visually illustrate the steps involved in processing the gathered data from Gridlab, Figure 6 is prepared. This figure provides a clear overview of the sequential actions and transformations that will be undertaken to analyze and prepare the data for the subsequent ML modelling process. It serves as a roadmap, guiding the researchers through the data preprocessing steps necessary for training and evaluating the ML model effectively. By systematically organizing the data into distinct input and output groups and outlining the steps involved in data processing through Figure 8, this research ensures a structured and methodical approach towards harnessing the insights hidden within the Gridlab data.

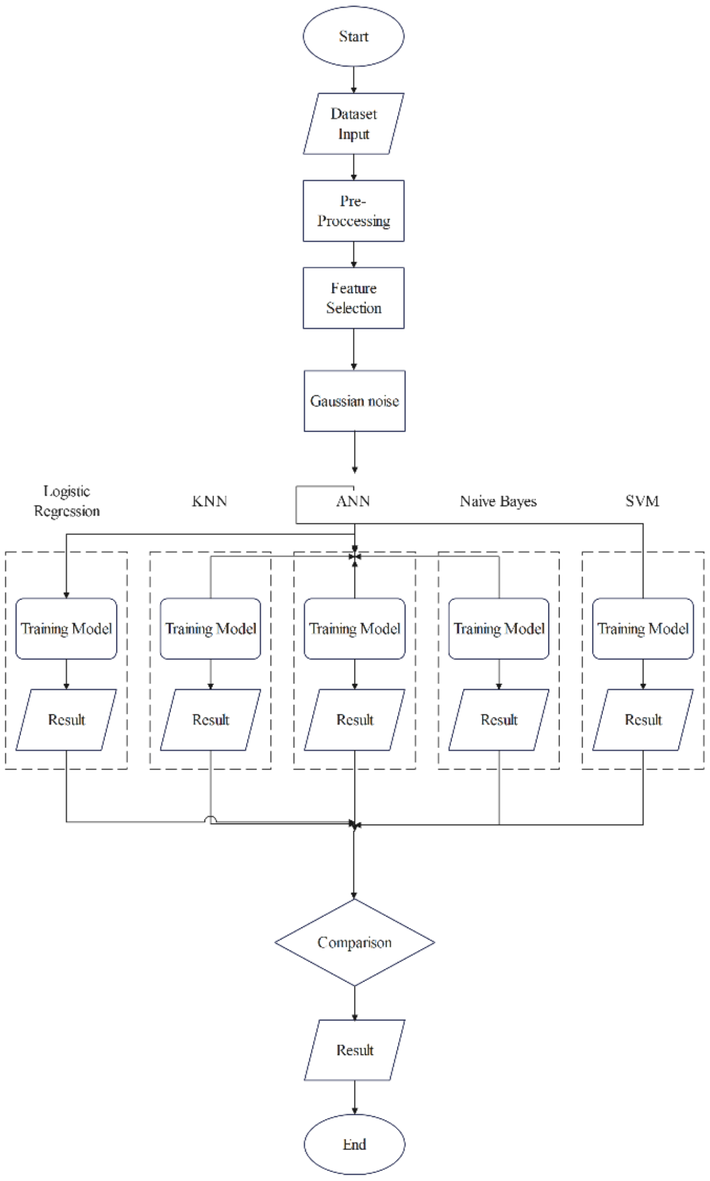


Figure 8 Ml techniques training steps

Chapter 5 : Results

This chapter presents the findings and analysis of the study on the prediction of EV states using ML methods. The study evaluates the impact of feature selection on the models' performance and considers correlation time and correlated noise to resemble real-life situations. Additionally, two datasets with distinct input scenarios were observed to ensure robustness. The data used in this research were obtained from GridLAB-D, encompassing power consumption information. Incorporating various scenarios as inputs, the ML models were trained and evaluated to predict the state of EVs. Five different ML methods were implemented using Python, and separate coding scripts were developed for each method, ensuring transparency and reproducibility. In evaluating the ML models, a range of evaluation methods was employed. The performance of the models was assessed using metrics such as ACC, F1 score, and MSE. These metrics were selected based on the nature of the prediction task and the target variable.

The study also considered correlation time to capture temporal dependencies in the data. By incorporating this aspect, the models could account for the time-varying nature of EV states and enhance the accuracy of predictions. Moreover, correlated noise resembled real-life scenarios, allowing the models to handle noise patterns commonly encountered in EV state data. To ensure the robustness of the results, two different datasets were employed, each representing distinct input scenarios. This approach enabled a comprehensive evaluation of the ML models' performance across various settings and provided a more thorough understanding of their effectiveness.

This chapter explained the coding scripts, highlighting each ML method's key steps and parameters. The results obtained from the evaluation metrics were thoroughly discussed, shedding light on the strengths and weaknesses of each model and allowing meaningful conclusions to be drawn regarding their performance.

The findings of this research contribute to the field of EV state prediction by demonstrating the impact of feature selection on model performance. Considering correlation time and correlated noise enhances the models' ability to capture real-life dynamics. Utilizing two different datasets with distinct input scenarios ensures the robustness of the results.

5.1 Grid Power Consumption

In this research, the grid power consumption data for analysis was specifically obtained from the NREL for the region of California, USA. California was selected as the focal point due to its prominent role in renewable energy initiatives and the widespread adoption of EVs. To capture diverse patterns and scenarios, the study examined two distinct periods. The first period ranged from 3 PM to 12 AM, encompassing the afternoon and evening hours, and the second period covered the duration from 12 AM to 6 AM, representing the late-night and early-morning periods.

These specific periods were chosen strategically to align with EVs' charging behaviour and availability in residential areas. During these periods, residential EV charging activities tend to be more prevalent as individuals return home from work or engagements, taking advantage of off-peak electricity rates. By focusing on these particular time spans, the research aimed to provide insights into the practical usage patterns and the impact of EV charging on grid power consumption in residential areas. This approach allowed for a comprehensive understanding of the interaction between EVs and the power grid during hours when EV charging activities are most prominent. Furthermore, considering these time spans facilitated the evaluation of the feasibility and effectiveness of implementing EV charging strategies during these specific hours. It enabled the researchers to assess the implications of EV charging on grid power consumption and explore opportunities for optimizing the integration of EVs into the existing power infrastructure.

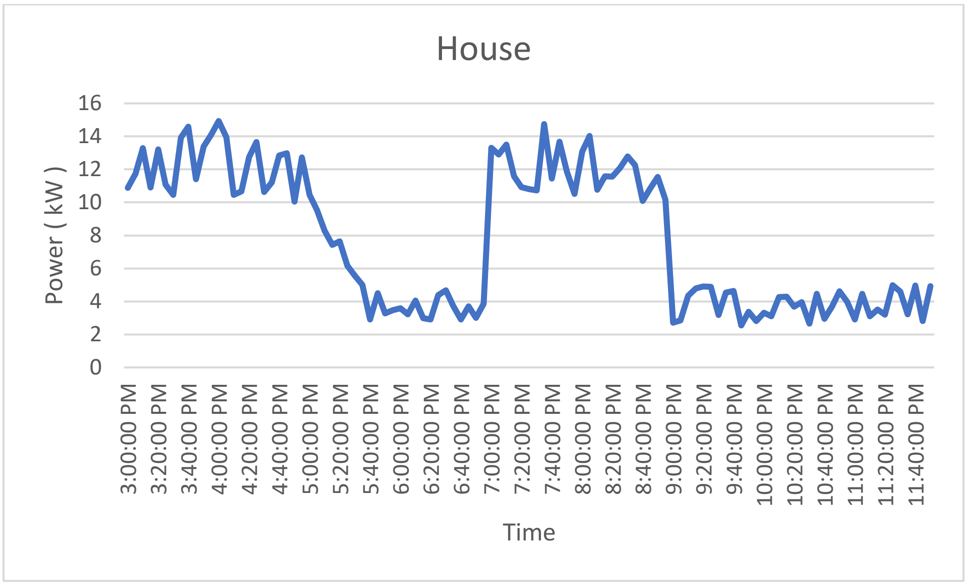


Figure 9 Household power consumption and v2g activation zone 3 pm to 12 am

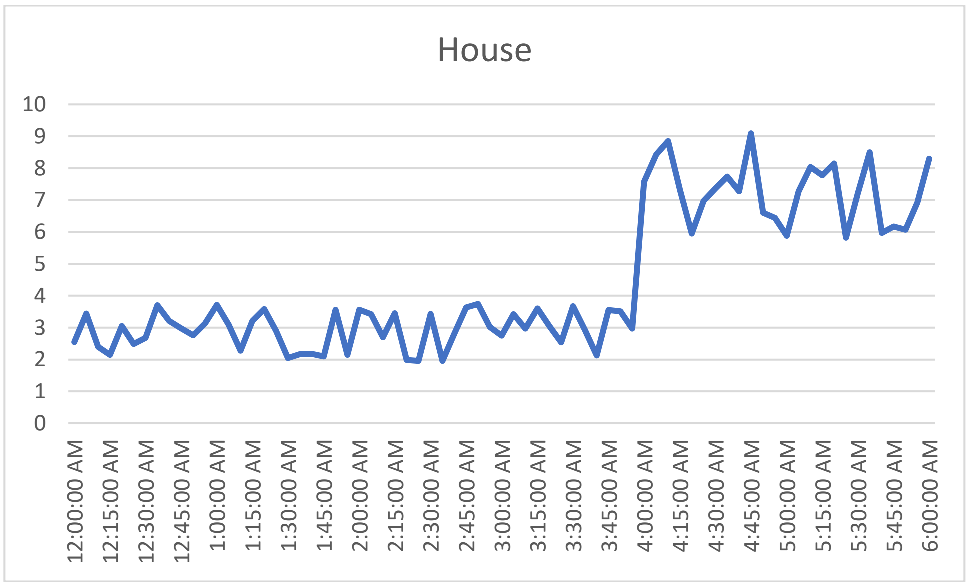


Figure 10 Household power consumption and v2g activation zone 12 am to 6 am

The first data set represents the power consumption for a typical day from 3:00:00 PM to 11:59:59 PM. The load consumption values seem to follow a relatively consistent trend during various times of the evening and night. Notably, there are two periods of peak load, one from 5:00 PM to 7:00 PM and another from 9:00 PM to 11:00 PM. These peak load periods indicate higher energy consumption, potentially coinciding with residents returning home and utilizing more appliances and lighting during the early evening hours.

The second data set represents the power consumption for a typical day from 12:00:00 AM to 6:00:00 AM. The load consumption shows a different pattern compared to the earlier hours. The load is generally lower during the late night and early morning hours, with a slight increase around 4:00 AM. This might correspond to activities such as overnight charging of devices or appliances running in standby mode.

Both data sets exhibit consistency and regularity in the power consumption pattern. The consumption during the late-night hours is generally lower, indicating reduced activity and usage of energy-intensive devices. The peak load periods in the evening suggest a higher demand for electricity as residents engage in various activities at home.

In this research, the default threshold assumption classified EVs with a SOC above 50% as "Idle," meaning they were not actively engaged in V2G or G2V operations. However, Figure 10 reveals an interesting observation during nighttime, where a small portion of time exhibits power consumption surpassing the predefined thresholds, indicating that some EVs are still charging. Notably, these instances are exceptions and not as prevalent as the daytime charging patterns.

Nevertheless, it is worth highlighting that allowing EVs to continue charging during nighttime aligns with maximizing EV capacity for users. The focus is to ensure that users have their EVs fully charged during periods of low electricity demand, optimizing their vehicle's range and usability throughout the day. Occasional instances where power consumption temporarily exceeds the thresholds can be attributed to individual user preferences or specific charging requirements. However, these occurrences are infrequent and do not significantly impact the overall charging strategy during nighttime hours.

Considering these factors, it is reasonable to continue charging all EVs during nighttime, even if, occasionally, power consumption surpasses the predefined thresholds. This approach enables users to utilize the capacity of their EVs fully and ensures their vehicles are adequately charged for their daily commuting needs, enhancing convenience and usability.

5.1.2 Owner Choice

2 Ev owner choice

Owner Number	Owner Choice – First one	Owner Choice – Second one
#1	**1**	**1**
#2	**1**	**-1**
#3	**1**	**0**
#4	**-1**	**1**

One part of this thesis examines EV owners' choices regarding bidirectional charging. This setting allows owners to determine how their EV participates in the system by selecting one of three numbers. The first option is 1, which means the EV can act as a V2G or G2V charger. The second option is -1, indicating that the vehicle should only be charged or G2V and not participate in V2G. The third option is 0, meaning the EV is disconnected from the system and idle. This setting can also be changed during the simulation for the owner. This declaration was the author's innovation to have a way to resemble the owner’s choice and show its effect on the system function.

5.1.3 SOC, Grid consumption, and V2G/G2V detection

In this thesis, the investigation focused on exploring the relationship between SOC, grid power consumption, time, and the detection of V2G or G2V modes in EVs. The main objective was to demonstrate scenarios where grid power consumption exceeds 9 kW, as indicated by the blue region in the diagram. Within these specific timeframes, when the battery SOC is above 70 percent, and the EV owner has chosen to engage V2G mode, represented by the grey area, the diagram captures and reflects this interaction. Four different EV cars with different aspects are considered for this work:

- **Tesla Model S**: The Tesla Model S is a luxury electric sedan known for its great performance and range. The battery capacity of the Model S varies depending on the model and version, but it typically ranges from around 75 kWh to 100 kWh.
- **Nissan Leaf**: The Nissan Leaf is a popular compact EV known for its practicality and affordability. The latest models of the Leaf come with a battery capacity of 40 kWh, providing a respectable range for daily commuting and urban driving.
- **Chevrolet Bolt EV**: The Chevrolet Bolt EV is a compact hatchback that offers a good balance of range and affordability. It has a battery capacity of 66 kWh, allowing for an estimated range of over 250 miles on a full charge.
- **Audi e-tron**: The Audi e-tron is a premium electric SUV that combines luxury with all-electric driving. It has a battery capacity of 95 kWh, providing a range of around 200 miles on a single charge.

By encapsulating SOC, grid power consumption, and the V2G/G2V detector in a single diagram, the aim was to emphasize their interdependency and crucial role in determining the operational mode of EVs. The diagram provides an intuitive representation of how these factors influence and affect one another, facilitating a comprehensive understanding of the dynamics within the system.

Furthermore, the diagram effectively illustrates distinct periods when grid power consumption is high, visually highlighting the time intervals in which V2G mode is activated to support the grid. During these high-consumption periods, surplus energy from the EV battery is harnessed to alleviate strain on the grid, resulting in a more balanced and efficient energy distribution. Conversely, during normal or low grid power consumption, denoted as zero or idle mode, the EV remains disconnected from the grid, conserving its energy for personal use.

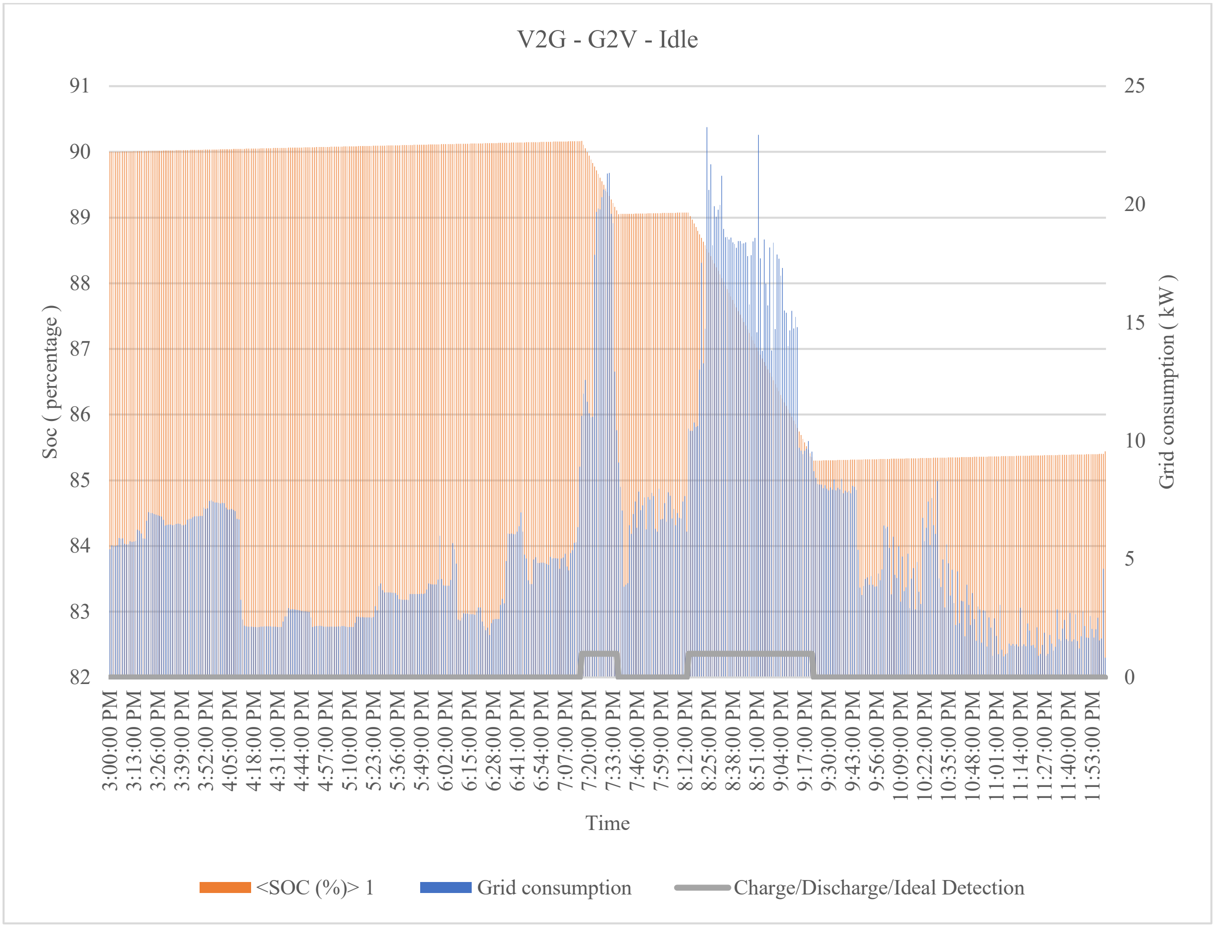


Figure 11 V2G - G2V - Idle for the first battery

The discharging section sheds light on a crucial aspect concerning the discharge rate of EV batteries, encompassing several key factors that merit closer examination:

- **Battery Chemistry**: EV batteries predominantly employ lithium-ion technology, and their chemical composition influences their discharge characteristics. At higher states of charge (SOC), the chemical reactions within the battery cells gradually slow down, resulting in a lower discharge rate. This limitation stems from the higher energy density associated with elevated SOC levels, where the battery's chemistry naturally restricts the rate at which energy can be released.
- **BMS**: The EV's BMS is critical in monitoring and controlling various aspects of the battery's operation, including the discharge rate. The BMS often regulates the discharging rate, particularly when the SOC is high, to safeguard the battery from over-discharge and ensure its

longevity. This protective measure prevents excessive stress on the battery cells and helps maintain their overall health and performance.

- **Performance Optimization**: EV manufacturers strive to strike a delicate balance between performance and battery longevity. By limiting the discharging rate at higher SOC levels, manufacturers aim to optimize the battery's overall performance and extend its lifespan. This approach ensures a consistent and reliable power output while minimizing the risk of degradation over time.
- **User Safety**: Ensuring user safety is paramount when determining the discharging rate. Restricting the discharging rate at higher SOC levels helps prevent situations where the battery rapidly depletes, potentially leaving the driver stranded without sufficient power. By controlling the discharging rate, EV manufacturers prioritize user safety and provide peace of mind to EV owners.

It is important to acknowledge that the specific discharging rate depicted in the diagram pertains to a single battery within the model. However, to enhance the credibility and reliability of the findings, it is crucial to incorporate randomization of discharging rates across multiple batteries in the model. This randomization accounts for the inherent variability observed in EVs equipped with different battery types, allowing for a comprehensive understanding of the system's overall performance. Considering the model's diverse range of EV batteries, a more realistic representation of real-world scenarios where EVs with different battery types coexist can be obtained. This approach enables the model to capture the inherent variability in discharging rates and provides a nuanced assessment of the system's behaviour.

Therefore, while the specific discharging rate illustrated in the diagram pertains to a single battery, it is imperative to emphasize the importance of randomizing discharging rates across multiple batteries within the model. This consideration ensures the generalizability and representativeness of the findings, accounting for the inherent variability observed in EVs equipped with different types of batteries. By incorporating this approach, the analysis gains credibility and offers valuable insights into the system's overall performance.

5.3.1.1 Household Power Consumption Variation

It could be seen in the previous figures that there is some abnormality in household power consumption during different times of the day. The observed grid power consumption data abnormalities can be attributed to various factors influencing electricity demand and usage patterns. These fluctuations in power consumption can arise due to the following reasons:

- **Usage Patterns**: The power consumption within a residential household can vary depending on the activities and behaviours of the occupants. Individuals or families have distinct routines, lifestyles, and energy consumption preferences. This can lead to fluctuations in power usage throughout the day, as different appliances and devices are turned on and off based on individual needs and preferences.
- **Seasonal Variations**: Power consumption in residential households can be influenced by seasonal variations. For instance, heating systems may be utilized more frequently during colder months, increasing power consumption. Similarly, air conditioning units or fans can increase energy usage in warmer months. These seasonal variations can lead to fluctuations in power consumption patterns.
- **Time of Day**: The time of day can also impact power consumption in residential households. During peak hours, such as early mornings when people wake up and evenings when they return home, higher power demands may be due to increased usage of appliances, lighting, and electronics. Conversely, during nighttime or off-peak hours, power consumption
- **Occupancy and Lifestyle**: The number of occupants and lifestyle choices within a residential household can affect power consumption. For example, a household with more occupants may have higher energy demands due to increased appliances, devices, and lighting usage.

Additionally, the residents' lifestyle choices, such as working from home, recreational activities, or hosting events, can contribute to fluctuations in power consumption.

- **Appliance and Equipment Efficiency**: The efficiency of appliances and equipment within a residential household can impact power consumption. Older or less energy-efficient appliances consume more power than newer, energy-efficient models. The presence of outdated or faulty equipment can lead to irregular power consumption patterns and potential spikes in energy usage.
- **Behavioural Factors**: Individual behaviours and habits of the occupants can influence power consumption. For instance, leaving lights, electronics, or appliances on when not in use or using high-energy-consuming devices excessively can contribute to abnormal power consumption patterns.
- **External Factors**: External factors such as weather conditions, voltage fluctuations, or power grid issues can also cause irregularities in power consumption data. Severe weather events, grid maintenance, or power outages can disrupt the normal power supply, resulting in fluctuations or inconsistencies in recorded power consumption.

It is important to consider that minute-by-minute data might exhibit more pronounced fluctuations than aggregated data over longer intervals. The granular nature of your data allows for a more detailed analysis of these variations, providing insights into the dynamic nature of power consumption patterns.

5.2 Machine Learning Results

This study is an in-depth investigation utilizing advanced machine learning techniques to analyze data collected from the Gridlab model. This research focused on the specific time range from 3 p.m. to 12 a.m., enabling the examination of diverse charging states of EVs resulting from various inputs and battery conditions. Also, the nighttime period from this analysis is excluded as it was assumed that all EVs were charging during that time. By delving into this time frame, valuable insights into the charging patterns of EVs and a comprehensive understanding of the dynamics of grid power consumption are gained. Two distinct datasets are utilised to ensure the reliability and accuracy of predictions are proper enough, each comprising 50,000 input data points. A robust analysis of the intricate relationship between input variables and the desired outcomes was conducted by harnessing such a large and diverse dataset. The objective was to enhance the generalizability and predictive capabilities of the machine learning models employed in this research. By incorporating multiple datasets and important data points, the aim is to gain profound insights into the complex interplay among SOC, grid power consumption, time, and the activation of V2G or G2V modes in EVs.

The primary aim of this research was to provide invaluable insights into EV behaviour, grid power consumption patterns, and the potential optimization of energy management strategies. By leveraging cutting-edge machine learning techniques and extensive datasets, the objective is to contribute to developing more efficient and sustainable energy systems, ultimately facilitating the seamless integration of EVs into the power grid. Gaussian Correlated Noises are introduced during the training process to assess the robustness of these trained models in the presence of unwanted noises. This enabled evaluation each method's performance and tolerance when confronted with unpredictable factors. It is important to note that the training data was carefully categorized based on relevant features. This categorization facilitated the models' ability to effectively capture and comprehend underlying patterns, thereby developing highly accurate and reliable predictions.

Various methodologies discussed in previous chapters to process the data are employed throughout the training phase. The ultimate objective was to optimize the models' performance and enhance their predictive capabilities. By training the models with meticulously organized and categorized data, The aim was to attain superior accuracy and gain profound insights into the research subject. The utilization of diverse training methods, incorporation of feature selection techniques, and comprehensive noise evaluation allowed for an exhaustive analysis of the data. The overarching goal was to develop robust models capable of accurately predicting EV behaviour, grid power consumption patterns, and activating V2G or G2V modes.

Table 3 General information about the ML data

Number of Data	**50,000 (For each)**
Train_test_split	**70% - 15% - 15%**
ML methods	**ANN – KNN – SVM – Naïve Bayes – Logistic Regression**
Evaluation methods	**ACC – MSE – F1Score**

The dataset was split into training and testing sets using the train_test_split method with a ratio of 70% for training, 15% for validation, and 15% for testing. This approach ensured that the models were trained on a significant portion of the data while also allowing for the evaluation of unseen data to assess their generalization performance. Various machine learning methods were employed, including ANN, KNN, SVM, Naïve Bayes, and Logistic Regression. Each method offers distinct advantages and may excel in different scenarios, providing comprehensive data analysis.

Splitting a dataset into training, validation, and testing sets is a common practice in machine learning to assess the performance of a model accurately and ensure its generalization to unseen data. The specific ratios you mentioned, 70% for training, 15% for validation, and 15% for testing, serve several important purposes:

- **Training Set (70%):**
  - The largest portion of the dataset is allocated for training the model. The model learns patterns, relationships, and features from this data. With more data available for training, the model can capture a wide range of variations in the input data and learn more robust representations.
- **Validation Set (15%):**

- The validation set tunes the model's hyperparameters and decides its architecture. The model is evaluated on the validation set after each epoch or iteration during training. This helps in monitoring the model's performance and preventing overfitting. Adjustments can be made based on validation performance to prevent the model from learning noise in the training data.

- **Testing Set (15%):**
  - The testing set is reserved for the final evaluation of the model's performance after training and hyperparameter tuning. By evaluating the model on unseen data, you can assess its ability to generalize to real-world scenarios. This evaluation provides a more accurate estimate of the model's performance in practice.

The chosen dataset split ratio serves multiple essential purposes. Firstly, dedicating a substantial portion to training enables the model to grasp intricate relationships, effectively capturing nuanced data patterns. Secondly, the allocation for validation is pivotal for hyperparameter tuning, allowing experimentation with different hyperparameter values to optimize the model's performance parameters, such as learning rate and regularization strength. Thirdly, the segregation of a distinct testing set ensures unbiased evaluation; this prevents potential bias if the same data is employed for training and testing. Lastly, the testing set is a benchmark to assess the model's ability to generalize to new, unseen data, showcasing its proficiency in recognizing meaningful patterns rather than merely memorising training data. This approach guarantees a well-rounded evaluation while fostering robust and dependable model development[64].

Three evaluation metrics were utilised to evaluate the performance of the trained models: ACC, MSE, and F1 Score. ACC measures the overall correctness of the predictions, while MSE quantifies the average squared difference between predicted and actual values. F1 Score combines precision and recall to evaluate the models' ability to classify the data accurately.

By employing multiple ML methods and utilizing different evaluation metrics, this research aimed to comprehensively assess the performance of the trained models and select the most suitable one for the given dataset and research objectives. The results obtained from the evaluation process would provide valuable insights into the effectiveness and efficiency of each method, enabling researchers and practitioners to make informed decisions regarding the selection and implementation of the most appropriate ML algorithm for similar EV integration studies.

5.2.1 Artificial Neural Networks

The model will be built by utilizing the following Python code snippets and implementing them in an ANN framework. This section will be discussed in detail to provide a comprehensive understanding of the Python code and the libraries that will be utilized.

5.2.1.1 NumPy

NumPy serves as the foundation of Python's scientific computing ecosystem. It is a powerful Python library that provides a multidimensional array object and various derived objects, including masked arrays and matrices. NumPy also offers a wide range of functions for efficient array operations, such as mathematical and logical operations, shape manipulation, sorting, selection, input/output operations, discrete Fourier transforms, elementary linear algebra, elementary statistical operations, random simulation, and much more [65].

5.2.1.2 Pandas

Pandas, a widely-used Python library, is an invaluable tool for data science, data analysis, and machine learning projects. It builds upon NumPy and facilitates working with arrays of different sizes. Pandas are commonly included in Python distributions, whether from operating systems or commercial vendor distributions like ActiveState's ActivePython, as a versatile data manipulation package. Pandas seamlessly integrate with other data science modules in the Python ecosystem [66].

5.2.1.3 Scikit-learn

Scikit-learn is an essential machine-learning package for the Python programming language. It provides various mathematical, statistical, and general-purpose algorithms that form the backbone of various machine-learning techniques. As an open-source library, Scikit-learn is crucial in creating and implementing various algorithms for machine learning and related technologies [67]. In the forthcoming code snippets, you will come across the "train_test_split" command, which divides the input data into specified ratios, allocating a portion for training and another for testing. Additionally, the code includes a step where the data is standardized. This process ensures that variables measured on different scales do not disproportionately influence the model fitting and learning process, thus mitigating potential bias. Standardizing features before model fitting is a common practice to address this concern.

By incorporating these methodologies and leveraging the power of Python libraries such as NumPy, Pandas, and Scikit-learn, this research aims to harness the capabilities of Artificial Neural Networks for robust and efficient modelling. The combination of these tools and techniques allows for advanced data manipulation, model training, and evaluation, ultimately contributing to the advancement of machine learning and its application in various domains.

5.2.1.4 Matplotlib

Matplotlib, a comprehensive visualization toolkit for Python, enables the creation of static, animated, and interactive visualizations. It empowers users to accomplish complex visualizations effortlessly and simplifies basic visualization tasks [68]. In the upcoming code segment, Matplotlib will be utilized to visualize the accuracy and loss of the prediction model as it progresses through the neural network learning process.

5.2.1.5 Keras

Keras, an API designed with human usability, emphasizes simplicity and ease of use. It follows best practices to reduce cognitive load, offering consistent and intuitive APIs and minimizing the number of user actions required for typical use cases. Keras is accompanied by extensive documentation and developer tutorials [69].

Keras was chosen for building and training ANN, specifically due to its specialized focus on deep learning architectures. A high-level and user-friendly interface is provided by Keras, which simplifies the processes of designing, configuring, and training complex neural networks. Through the exclusive utilization of Keras for ANNs, access was gained to its extensive library of pre-designed layers and activation functions, allowing for the seamless construction of intricate network structures. Keras' compatibility with various backends, such as TensorFlow, ensures optimal performance during training. Moreover, Keras streamlines model evaluation by integrating with the broader Scikit-learn ecosystem, allowing for consistent metric calculation and model assessment across the project. This strategic decision allowed this procedure to effectively leverage Keras' deep learning capabilities, concentrating efforts on creating robust and accurate neural network models for my specific task while maintaining a streamlined development process.

5.2.1.6 Python Coding

The initial section of the code imports several libraries and packages, including numpy, pandas, matplotlib, stats models, and sci-kit-learn. Additionally, it employs the read_excel() function from the pandas' module to read an Excel dataset, storing it in a pandas DataFrame named 'dataset.' To ensure reproducibility, the code shuffles the rows of the dataset using the sample() function with a random_state parameter of 2. The shuffled dataset is then assigned to a new data frame called 'shuffled_df.' Subsequently, a subset of 20,000 randomly selected rows from 'shuffled_df' is assigned to another data frame named 'dataset2,' achieved by utilizing the sample() function with the parameter n=50000. This code snippet proves useful for data preparation in machine learning tasks, as it shuffles the data and selects a subset for model training and testing. The StandardScaler class from scikit-learn can normalize the dataset, while the train_test_split() function facilitates splitting the dataset into training and testing sets. Furthermore, the statsmodels package can be leveraged for constructing and evaluating statistical models.

By incorporating these tools and techniques, this research endeavours to employ Python's extensive ecosystem, encompassing Matplotlib, Keras, and other libraries, to facilitate data visualization, model training, and evaluation. Utilizing these resources contributes to developing robust machine-learning models and fosters comprehensive analysis and interpretation of the results within the research domain. Each technique and the related that are used for each one are mentioned in Figure 12.

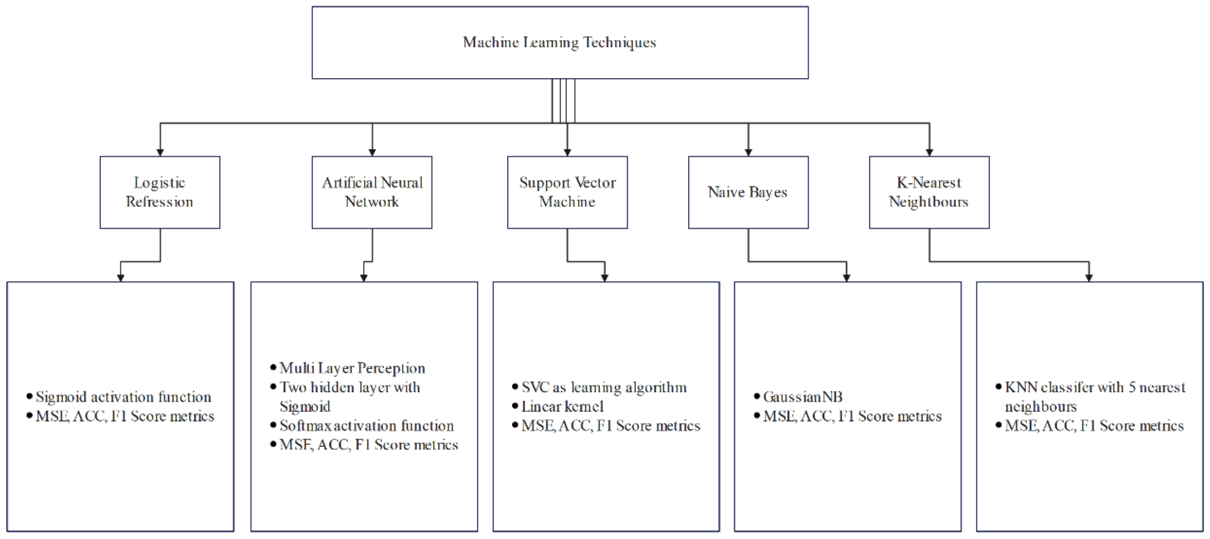

Machine Learning Techniques
Logistic Refression
Artificial Neural Network
Support Vector Machine
Naive Bayes
K-Nearest Neightbours
• Sigmoid activation function
• MSE, ACC, F1 Score metrics
• Multi Layer Perception
• Two hidden layer with Sigmoid
• Softmax activation function
• MSE, ACC, F1 Score metrics
• SVC as learning algorithm
• Linear kernel
• MSE, ACC, F1 Score metrics
• GaussianNB
• MSE, ACC, F1 Score metrics
• KNN classifer with 5 nearest neighbours
• MSE, ACC, F1 Score metrics

Figure 12 Machine learning techniques and their related toolkits

5.3 Feature Selection

In the subsequent section, three methods were employed to identify and eliminate features with minimal impact on the model. These methods include the Feature Correlation Matrix, Feature Impression

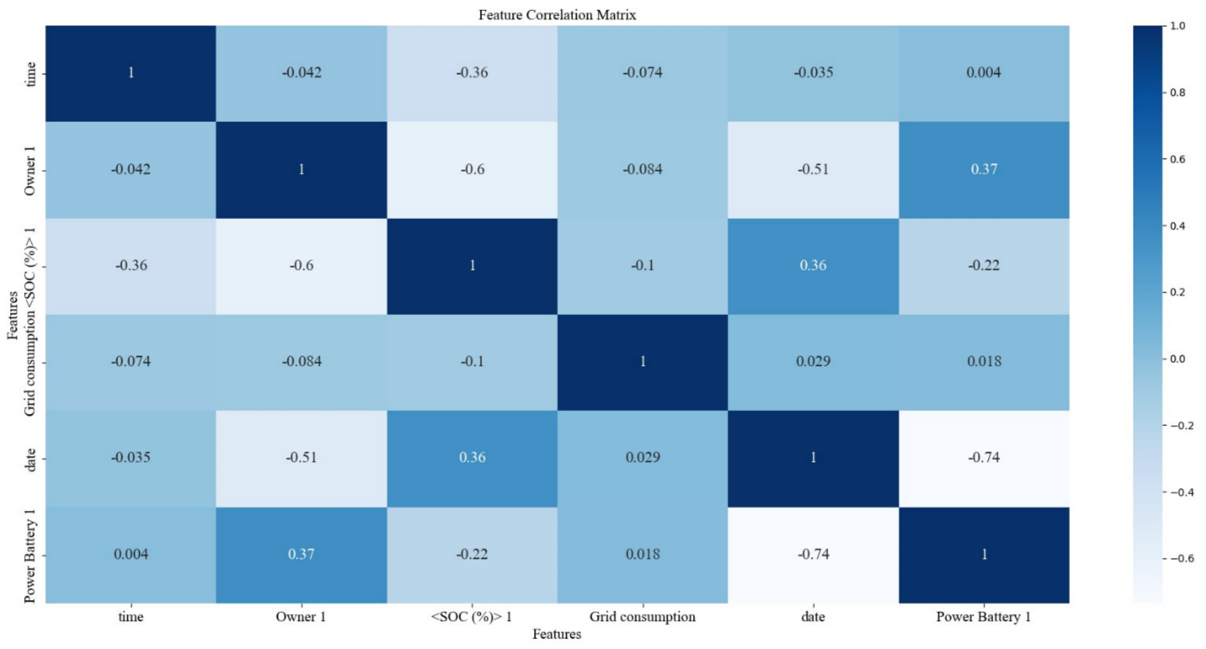
Ratings, and the Feature Accuracy Comparison Table. The first two methods were discussed in the preceding section, and in this part, the corresponding figures will be delved into.

To initiate the process, the Features Correlation Matrix was utilized. Figure 12 showcases the relationships between each variable and others. As depicted in Figure 7, all six features exhibit varying degrees of correlation. However, it is important to note that the correlation values do not indicate strong relationships that would necessitate the removal of any specific feature. The correlation values within the matrix range from -1 to 1, representing the strength and direction of the linear relationship between feature pairs. A correlation coefficient 1 signifies a perfect positive linear relationship, while -1 indicates a perfect negative one. Values near 0 imply a weak or no linear relationship between the features.

Based on the correlation matrix, it can be observed that the selected features display moderate correlations. This suggests that each feature contributes specific information to the dataset and is not redundantly correlated with others. Therefore, retaining all the features is advantageous, as they collectively represent the data comprehensively. By preserving all six features, the unique information contributed by each feature can be utilized by the model, resulting in more accurate and robust analysis. This approach acknowledges the individual contributions of the features while considering their collective impact on the research objectives. In addition to the feature correlation analysis, feature importance was assessed as part of the feature selection process. The most influential variables in predicting the desired outcome can be identified by evaluating the importance of each feature. Figure 14 presents the feature importance results, ranking the features based on their importance scores.

The analysis shows that Time, SOC, Grid power consumption, and Owner choice exhibit relatively higher importance scores than the other features. This indicates that these four features significantly contribute to determining the desired outcome or target variable. Conversely, Date and Power batteries demonstrate relatively lower importance scores, suggesting their limited contribution to the model's predictive power.

The importance scores in Figure 13 range from 0 to 1, with higher scores indicating greater importance. Features with higher scores exert a stronger influence on the outcome variable, and including them in the model can enhance prediction accuracy. Conversely, features with lower scores may have minimal impact on the outcome variable and can potentially be disregarded without compromising the model's predictive performance.



	time	Owner 1	<SOC (%)> 1	Grid consumption	date	Power Battery 1
time	1	-0.042	-0.36	-0.074	-0.035	0.004
Owner 1	-0.042	1	-0.6	-0.084	-0.51	0.37
<SOC (%)> 1	-0.36	-0.6	1	-0.1	0.36	-0.22
Grid consumption	-0.074	-0.084	-0.1	1	0.029	0.018
date	-0.035	-0.51	0.36	0.029	1	-0.74
Power Battery 1	0.004	0.37	-0.22	0.018	-0.74	1

Figure 13 Feature correlation matrix

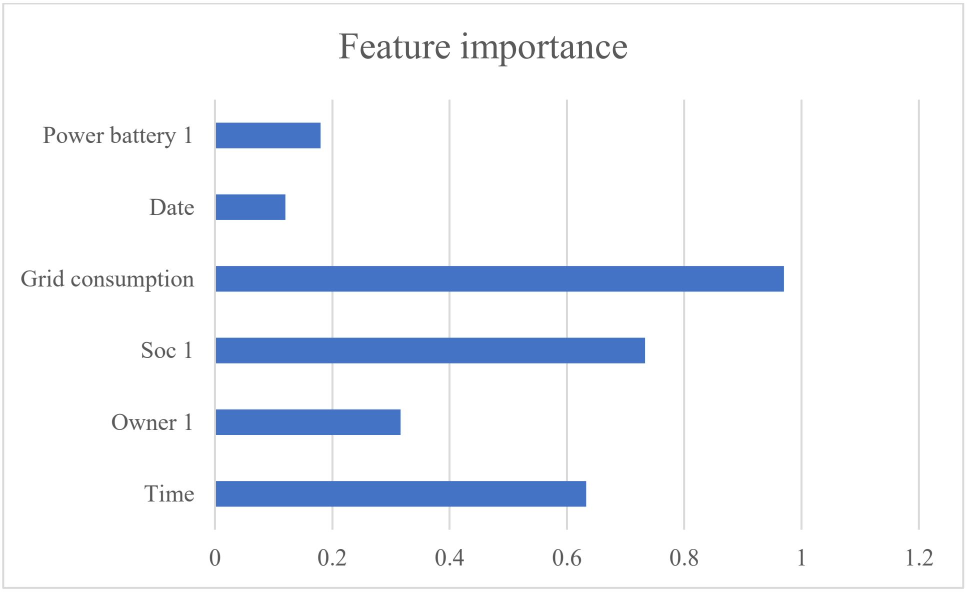


Figure 14 Feature impression rating

According to the last achievement, now it is time for the evaluation to see whether eliminating these features affects the accuracy of the ML techniques. For this test, all five methods are compared before and after the feature selection, which can be seen in the next tables.

4 Accuracy comparison before and after feature selection

ACC Comparison	ACC - Before	ACC - After
Logistic Regression	0.9514	0.9524
Naïve Bayes	0.5619	0.5694
KNN	0.9866	0.9871
SVM	0.9616	0.9611
ANN	0.9944	0.9954

5 F1-Score comparison before and after feature selection

F1-Score Comparison	F1-Score - Before	F1-Score - After
Logistic Regression	0.9439	0.9443
Naïve Bayes	0.5716	0.5779
KNN	0.9866	0.9855
SVM	0.9611	0.9605
ANN	0.9944	0.9957

Table 6 Mse comparison before and after feature selection

MSE Comparison	MSE - Before	MSE - After
Logistic Regression	0.0686	0.0664
Naïve Bayes	0.533	0.5277
KNN	0.02	0.026
SVM	0.0725	0.075
ANN	0.0064	0.0052

After conducting a meticulous analysis and carefully evaluating the outcomes, a compelling conclusion is reached that employing feature selection techniques does not substantially impact this model's accuracy. Hence, a thoughtful decision has been made to include all the input features in the training process, ensuring that the model embraces the entirety of the data. By adopting this inclusive approach, all the available information can be comprehensively considered, and the risk of overlooking potentially valuable features is avoided.

Nevertheless, it is crucial to acknowledge that prior findings did not consider the notion of correlation over time. As this research progresses, the intention is to incorporate different correlation times to explore the effectiveness of feature selection across various temporal scales. By introducing correlation time as a factor, the aim is to investigate how the model's performance and predictive capabilities can be impacted by including or excluding specific features during specific time intervals.

Through this thorough examination with varying correlation times, invaluable insights are expected to be gained into the influence of feature selection within different temporal contexts. This will enable uncovering potential periods or patterns where certain features become more or less relevant in making accurate predictions. The insights can be enriched by understanding the intricate relationship between features, time, and their significance in predictive modelling.

Ultimately, the goal is to enhance the understanding of the data and fine-tune the model by exploring the effectiveness of feature selection across diverse scales while considering correlation time. This holistic and meticulous approach ensures that the research captures potential variations and provides robust and reliable insights into the intricate interplay between features, time, and the model's predictive performance. [70].

Table 7 Short correlation time - before feature selection - 5 evaluation

Short correlation time - Before feature selection - 5	**ACC**	**F1-Score**	**MSE**
Logistic Regression	**0.5469**	**0.4019**	**0.4457**
Naïve Bayes	**0.6525**	**0.4865**	**0.4735**
KNN	**0.5304**	**0.5611**	**0.5391**
SVM	**0.7003**	**0.6483**	**0.5971**
ANN	**0.7758**	**0.5938**	**0.4725**

Table 8 Medium correlation time - before feature selection - 15 evaluation

Medium correlation time - Before feature selection - 15	**ACC**	**F1-Score**	**MSE**
Logistic Regression	**0.5912**	**0.3776**	**0.4420**
Naïve Bayes	**0.6422**	**0.4996**	**0.4799**
KNN	**0.6291**	**0.5523**	**0.4581**
SVM	**0.5852**	**0.7000**	**0.5154**
ANN	**0.7172**	**0.4687**	**0.4277**

Table 9 Long correlation time - before feature selection - 60 evaluation

Long correlation time - Before feature selection - 60	ACC	F1-Score	MSE
Logistic Regression	**0.6445**	**0.3982**	**0.3520**
Naïve Bayes	**0.5709**	**0.5543**	**0.4709**
KNN	**0.5116**	**0.4976**	**0.5303**
SVM	**0.6054**	**0.6173**	**0.4072**
ANN	**0.6041**	**0.5051**	**0.4557**

Table 10 Short correlation time - after feature selection – 5 evaluation

Short correlation time - After feature selection - 5	ACC	F1-Score	MSE
Logistic Regression	**0.8941**	**0.878**	**0.1153**
Naïve Bayes	**0.63**	**0.5715**	**0.5341**
KNN	**0.8547**	**0.8475**	**0.1802**
SVM	**0.8864**	**0.8831**	**0.1474**
ANN	**0.9463**	**0.9469**	**0.0088**

Table 11 Short correlation time - after feature selection – 15 evaluation

Medium correlation time - After feature selection - 15	ACC	F1-Score	MSE
Logistic Regression	**0.8949**	**0.8789**	**0.1228**
Naïve Bayes	**0.5667**	**0.5708**	**0.5233**
KNN	**0.7422**	**0.7212**	**0.2794**
SVM	**0.8849**	**0.8818**	**0.1423**
ANN	**0.9559**	**0.9556**	**0.0675**

Table 12 Long correlation time - after feature selection – 60 evaluation

Long correlation time - After feature selection - 60	ACC	F1-Score	MSE
Logistic Regression	**0.8919**	**0.873**	**0.1304**
Naïve Bayes	**0.5667**	**0.5736**	**0.5269**
KNN	**0.6042**	**0.5598**	**0.443**
SVM	**0.8579**	**0.8516**	**0.1598**
ANN	**0.9491**	**0.9487**	**0.0799**

Based on the previous results that can be seen in Table 7 - 12, which investigated the effectiveness of feature selection using correlation time on an ML model, the results suggest that considering correlation time can improve the accuracy of the model, as short, medium, and long correlation times were found to be effective in selecting relevant features. This is because the selected features capture different time intervals during which they are correlated with the target variable. However, feature selection did not affect the model's accuracy when no correlation time was used. This indicates that the correlation between the features and the target variable depends on the time interval or lag between them. Therefore, failing to consider time lags can result in the lack of effect of feature selection. Furthermore, increasing the lagged steps in correlation time led to a decrease in accuracy after feature selection. This suggests that the correlation between the features and the target variable may not be significant or informative for the model beyond a certain time lag.

Since the prediction model also has to be effective during different correlation times, the outcome of this experiment will be using the feature selection and elimination of two unimportant features: Date and Power Battery Consumption. So, the final ML model can be seen in Figure 15.

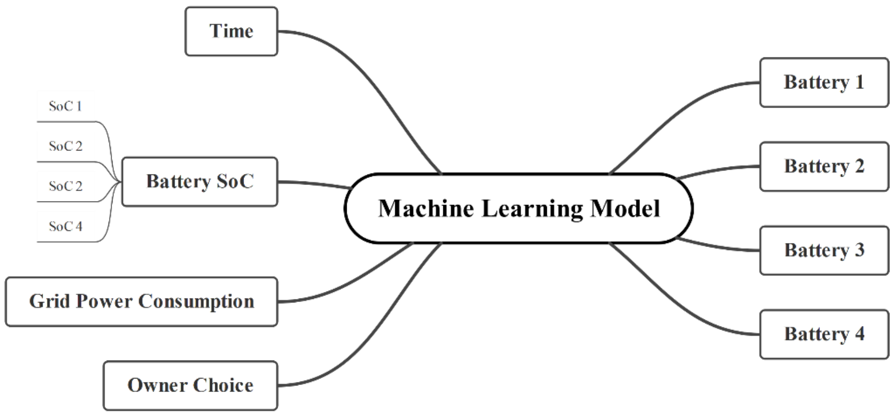


Figure 15 Modified ml model

5.4 Correlated Noise

As mentioned earlier, this thesis aims to assess the resilience of each technique against various types of noise that can potentially affect the accuracy of the data. To achieve this, Gaussian Correlated Noises will be introduced to the training data of the machine learning model. Among the six inputs, namely Battery Power Consumption, Grid Consumption, and Battery State of Charge (SOC), only these inputs with float data type will be subjected to noise injection. The accuracy of the model will then be evaluated using test data. This simulation will also incorporate Different noise intensity levels to ensure a thorough evaluation. No correlation time is used for this part, and different standard deviations (STD)s were applied after using the feature selection.

In Chapter 3 of the research, creating different noises with varying strengths involves adjusting the standard deviation (σ) parameter. The noise matrix used for training data is generated by manipulating this parameter to reflect a range of noise strengths. The aim is to explore the impact of different noise levels on the performance of the trained models.

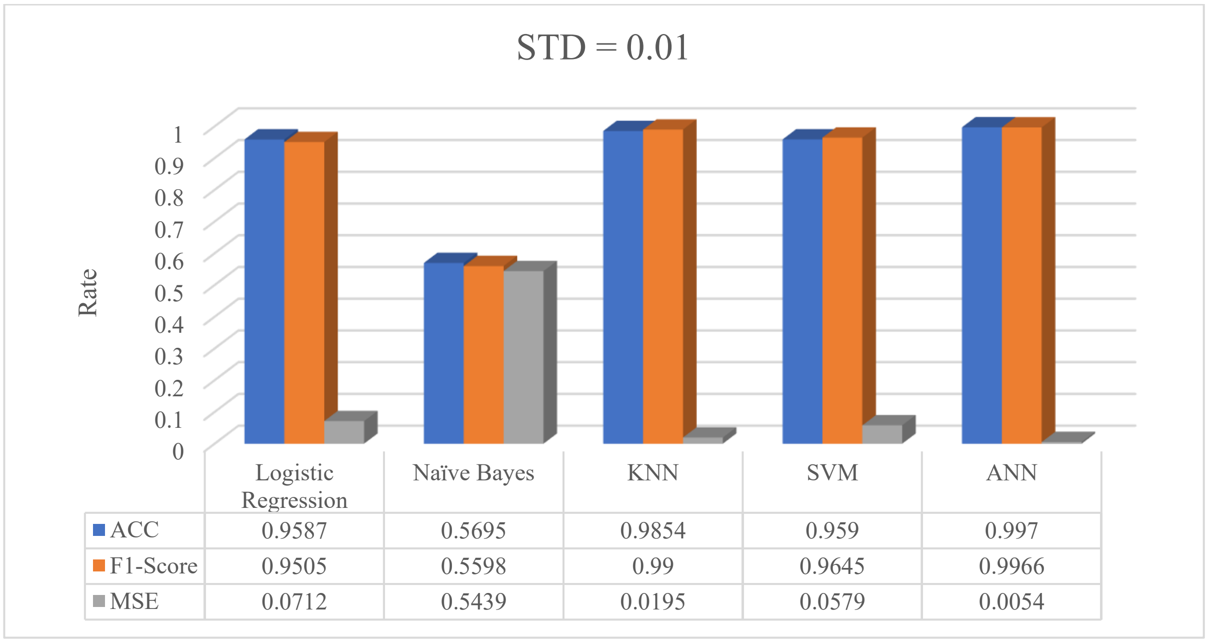
Subsequently, evaluation variables are employed to assess and compare the performance of the models under each noise condition. These evaluation variables serve as metrics for determining the effectiveness of each method in handling the noise and producing accurate predictions. By analyzing the results obtained from these evaluations, the research seeks to identify the best-performing method among the different noise scenarios.

In machine learning, robustness refers to a model's ability to perform consistently and accurately across varying conditions, including noisy or uncertain input data. A robust model demonstrates resilience against minor perturbations or deviations in the data, thus maintaining its predictive power even when the input is less than ideal. The concept of robustness is particularly relevant in real-world scenarios where data can be noisy, incomplete, or subject to unforeseen variations.

The impact of introducing controlled noise to the EV data was explored to address the concept of robustness in the research. The objective was to simulate realistic scenarios where input data might have inherent uncertainties or inaccuracies adding noise to the EV-related features; the aim was to assess the extent to which the stability and accuracy of the model's predictive capabilities remain intact in the presence of such variations Essentially, the idea was to ensure that the model's performance does not deteriorate significantly under circumstances that might challenge the precision of its predictions.

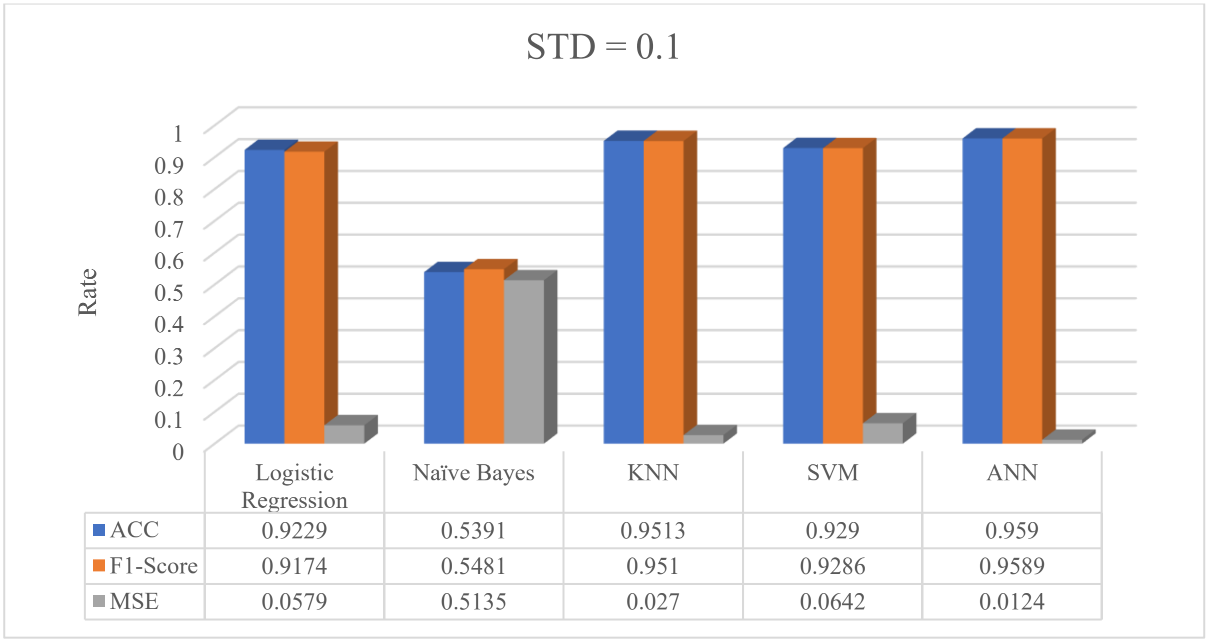
Incorporating noise into the EV data served to assess the model's resilience and adaptability. If the model demonstrated consistent and accurate predictions despite the introduced noise, it would suggest a higher level of robustness. This enhanced robustness could improve performance in real-world scenarios where data imperfections are common. By exploring the concept of robustness and its application in the context of noise-induced variations, my research aimed to contribute to the development of more reliable and effective predictive models in the domain of EV battery state forecasting and energy management.

To ensure the reliability of the conclusion, two different datasets with various initial conditions are used. By comparing the results of these two datasets, it can be ensured that the best technique is the right choice and can be utilized for further development.



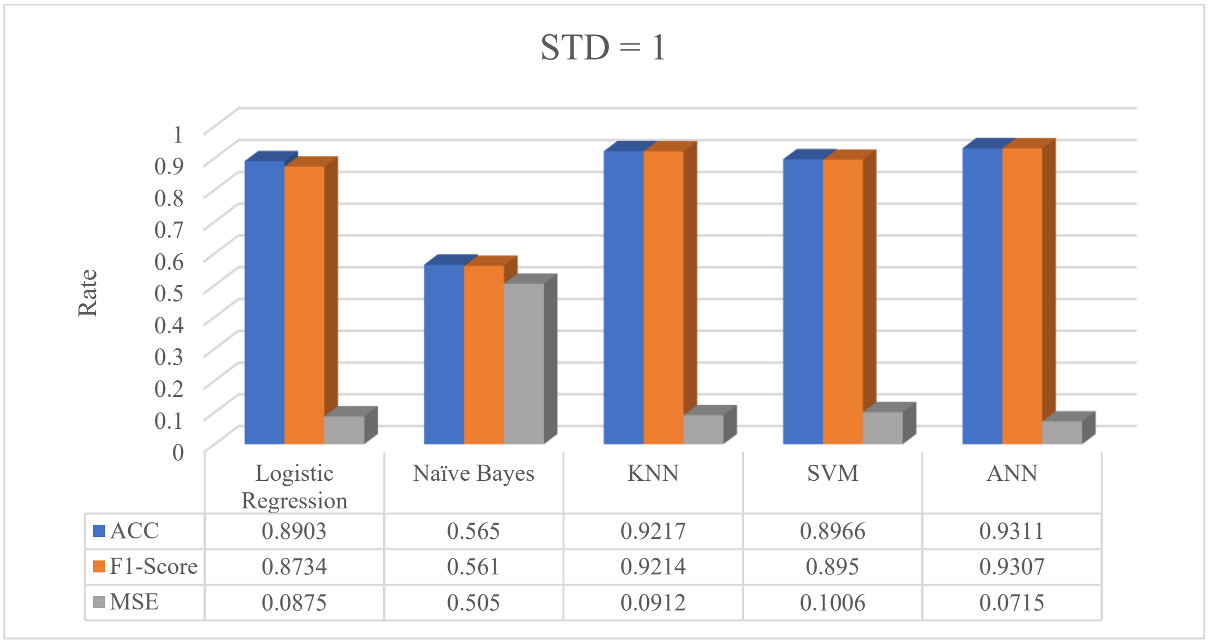
	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
🟦 ACC	0.9587	0.5695	0.9854	0.959	0.997
🟧 F1-Score	0.9505	0.5598	0.99	0.9645	0.9966
⬜ MSE	0.0712	0.5439	0.0195	0.0579	0.0054

Figure 16 Gaussian correlated noise with std of 0.01 for the first dataset



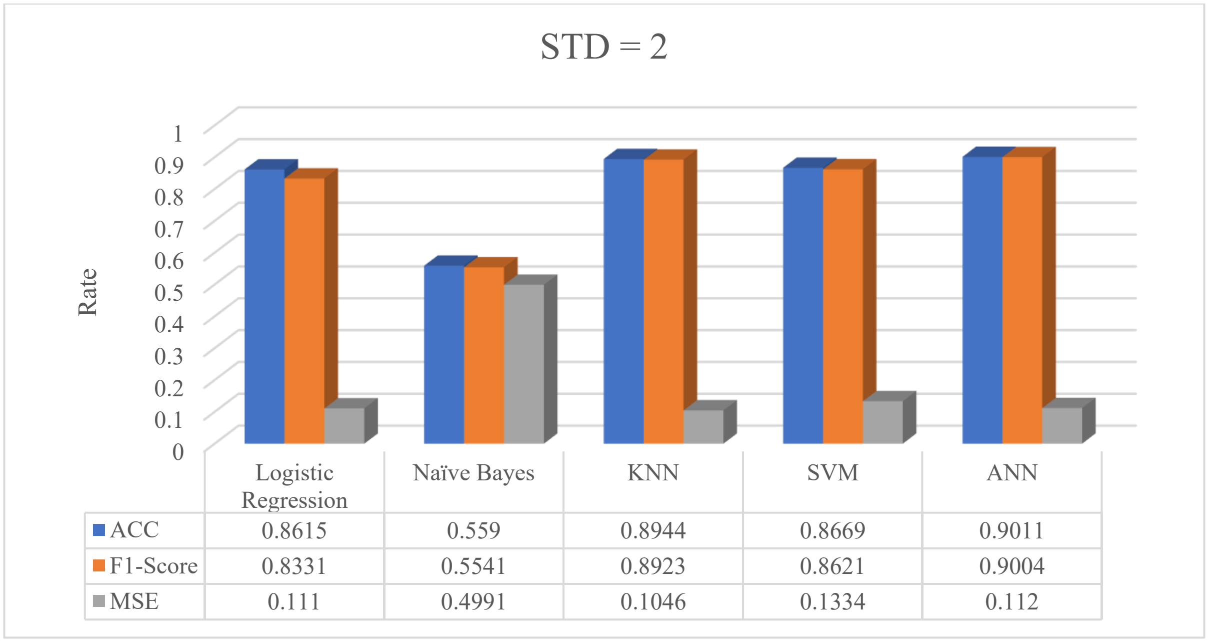
	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
🟦 ACC	0.9229	0.5391	0.9513	0.929	0.959
🟧 F1-Score	0.9174	0.5481	0.951	0.9286	0.9589
⬜ MSE	0.0579	0.5135	0.027	0.0642	0.0124

Figure 17 Gaussian correlated noise with std of 0.1 for the first dataset



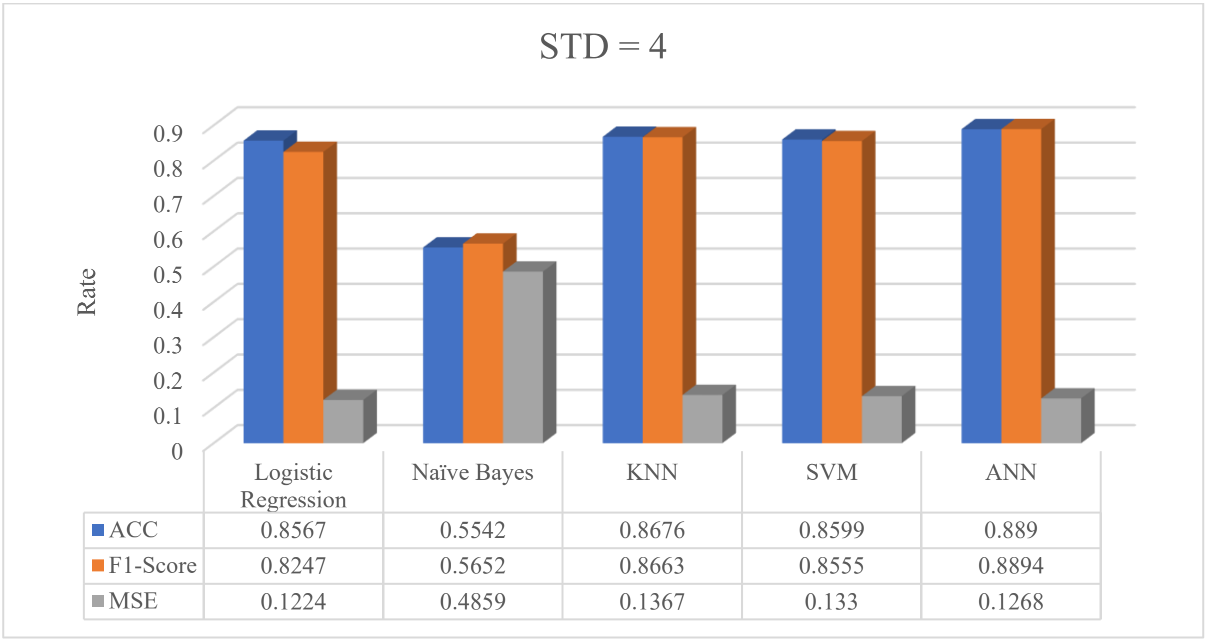
	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
■ ACC	0.8903	0.565	0.9217	0.8966	0.9311
■ F1-Score	0.8734	0.561	0.9214	0.895	0.9307
■ MSE	0.0875	0.505	0.0912	0.1006	0.0715

Figure 18 Gaussian correlated noise with std of 1 for the first dataset



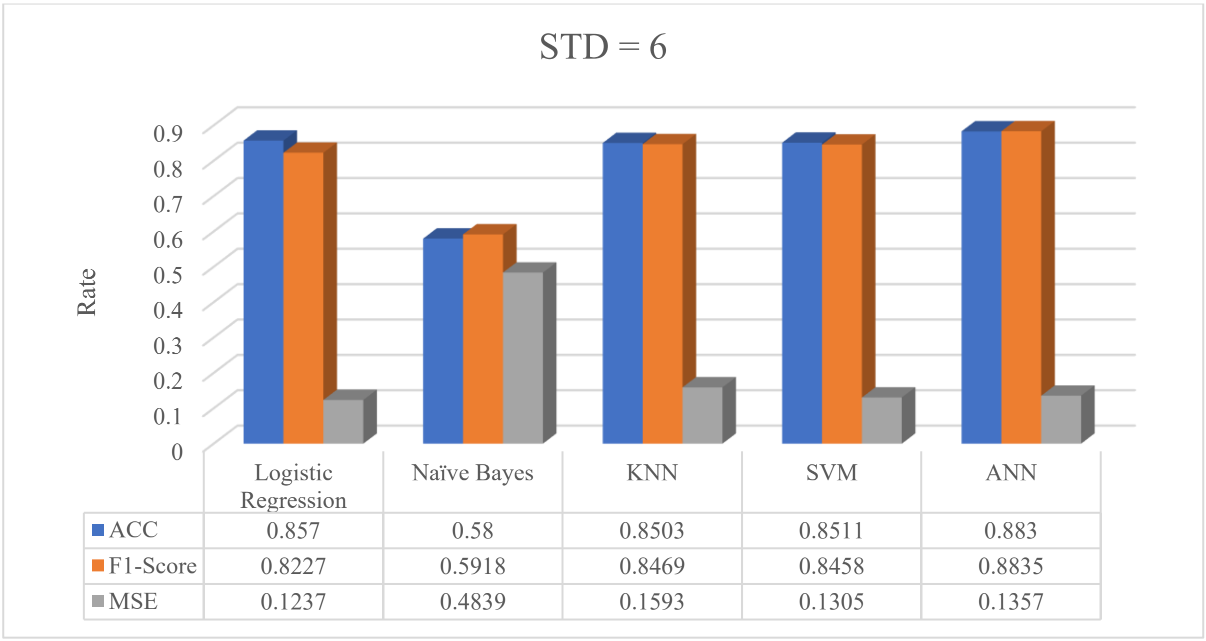
	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
■ ACC	0.8615	0.559	0.8944	0.8669	0.9011
■ F1-Score	0.8331	0.5541	0.8923	0.8621	0.9004
■ MSE	0.111	0.4991	0.1046	0.1334	0.112

Figure 19 Gaussian correlated noise with std of 2 for the first dataset



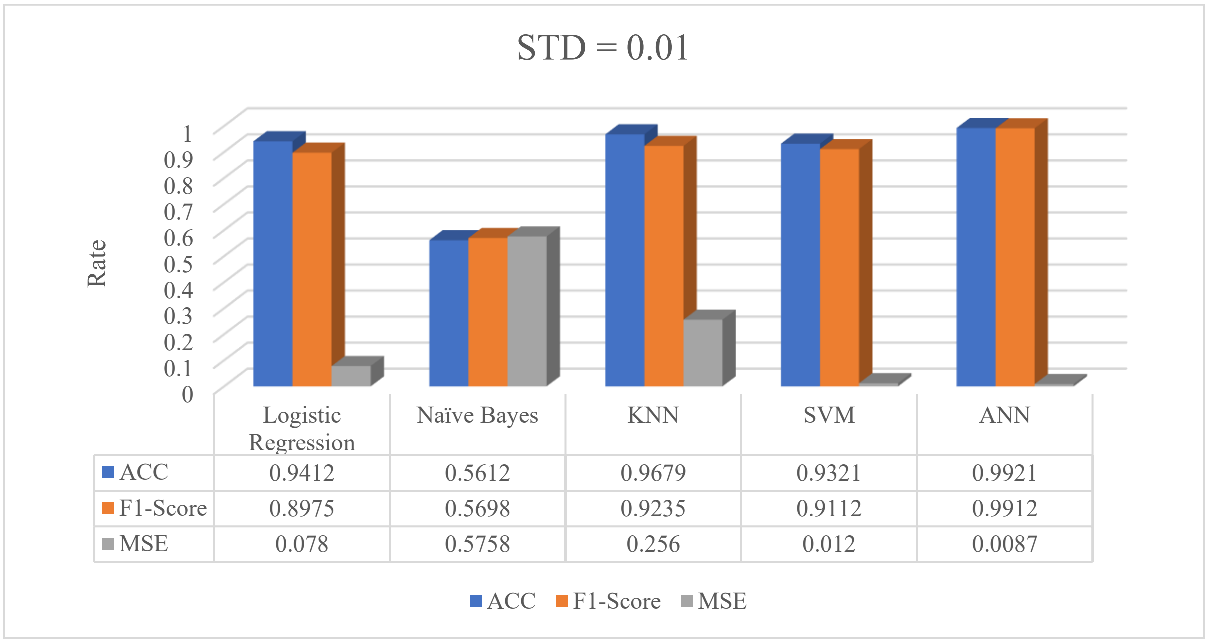
	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
🟦 ACC	0.8567	0.5542	0.8676	0.8599	0.889
🟧 F1-Score	0.8247	0.5652	0.8663	0.8555	0.8894
◼ MSE	0.1224	0.4859	0.1367	0.133	0.1268

Figure 20 Gaussian correlated noise with std of 4 for the first dataset



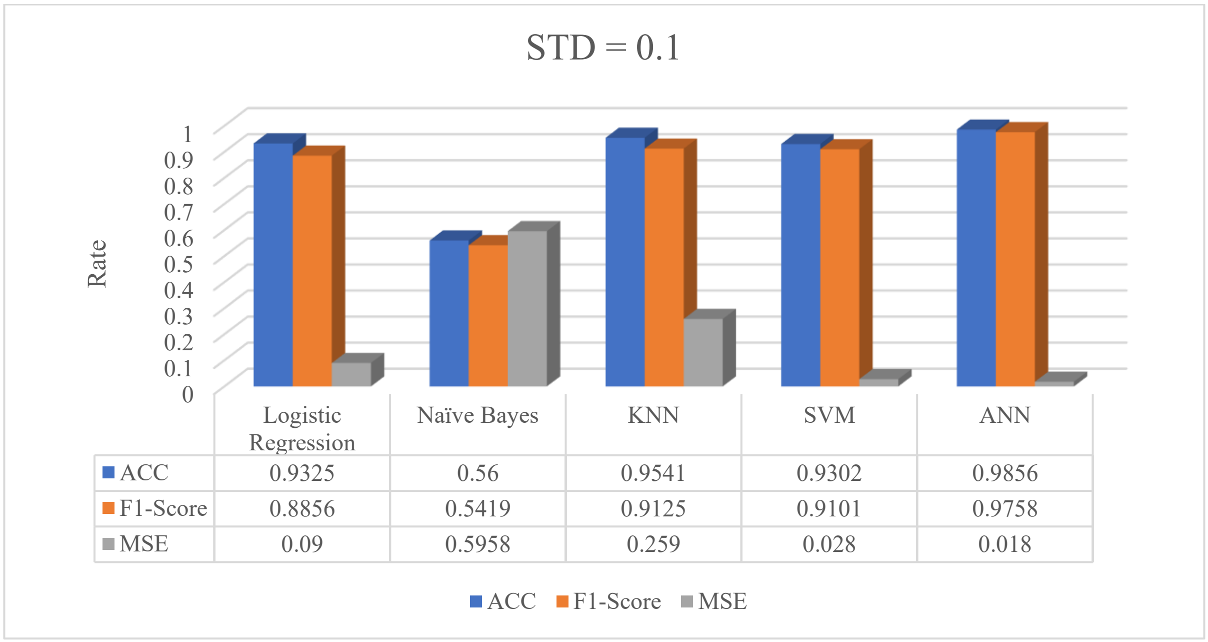
	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
🟦 ACC	0.857	0.58	0.8503	0.8511	0.883
🟧 F1-Score	0.8227	0.5918	0.8469	0.8458	0.8835
◼ MSE	0.1237	0.4839	0.1593	0.1305	0.1357

Figure 21 Gaussian correlated noise with std of 6 for the first dataset



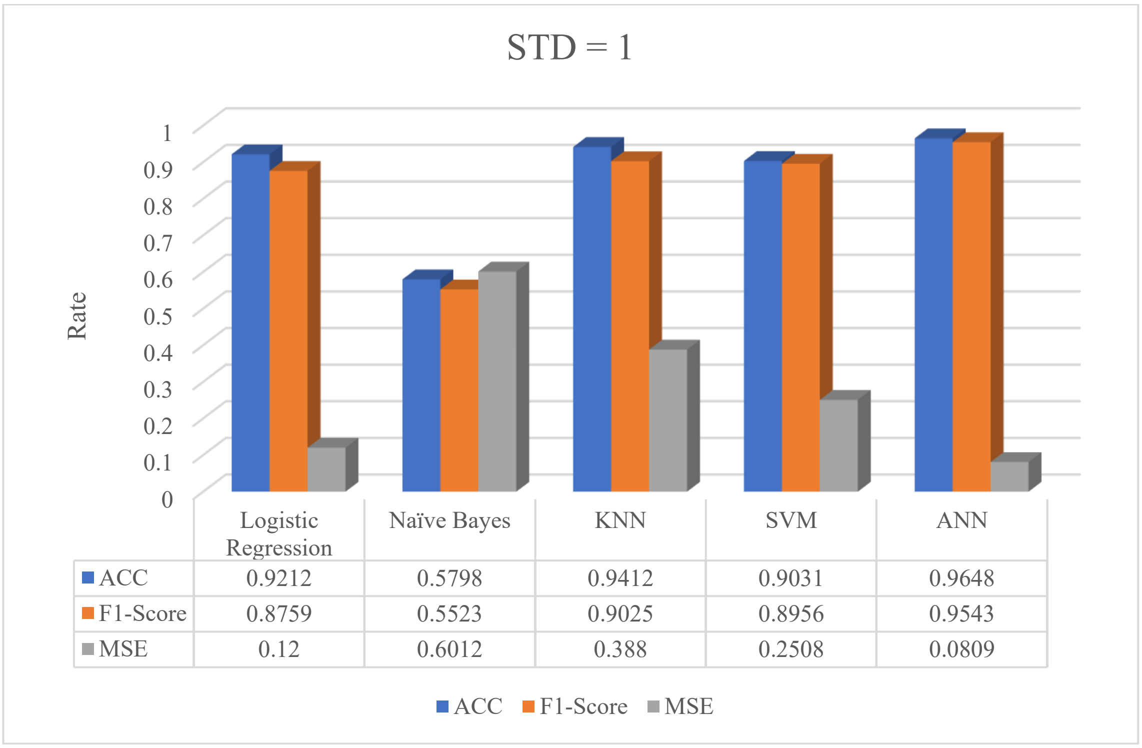
	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
🟦 ACC	0.9412	0.5612	0.9679	0.9321	0.9921
🟧 F1-Score	0.8975	0.5698	0.9235	0.9112	0.9912
⬜ MSE	0.078	0.5758	0.256	0.012	0.0087

Figure 22 Gaussian correlated noise with std of 0.01 for the second dataset



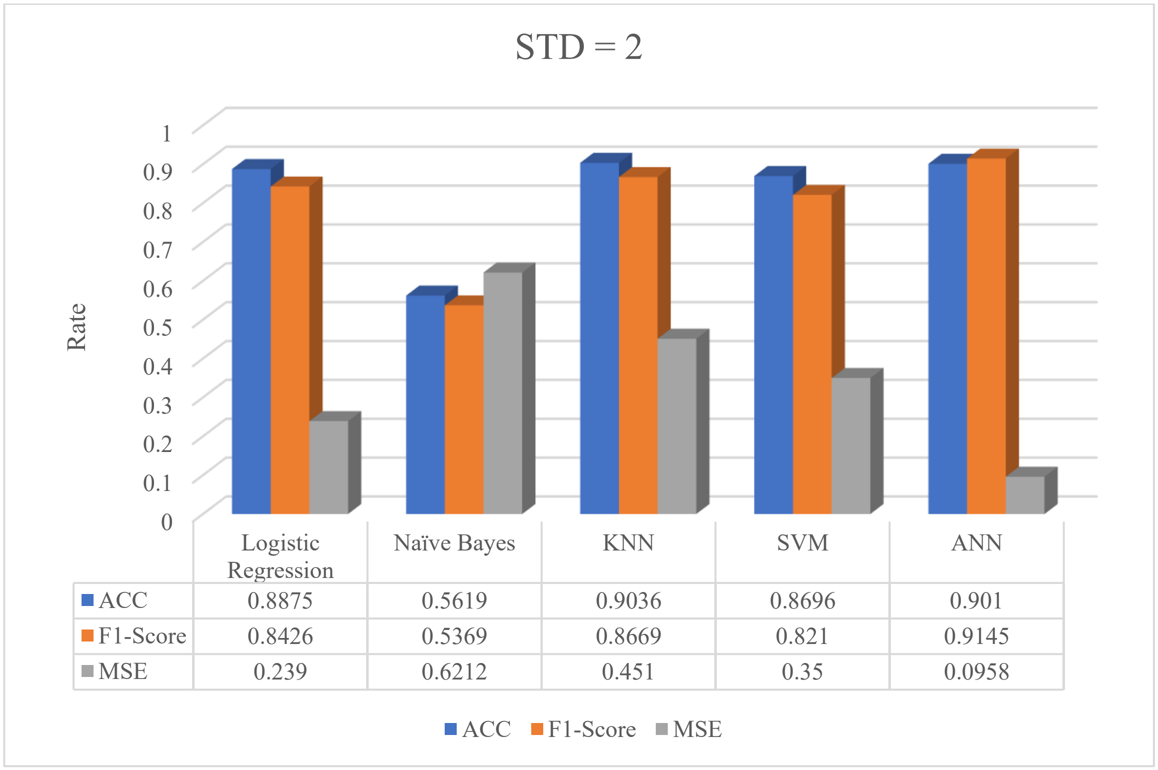
	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
🟦 ACC	0.9325	0.56	0.9541	0.9302	0.9856
🟧 F1-Score	0.8856	0.5419	0.9125	0.9101	0.9758
⬜ MSE	0.09	0.5958	0.259	0.028	0.018

Figure 23 Gaussian correlated noise with std of 0.1 for the second dataset



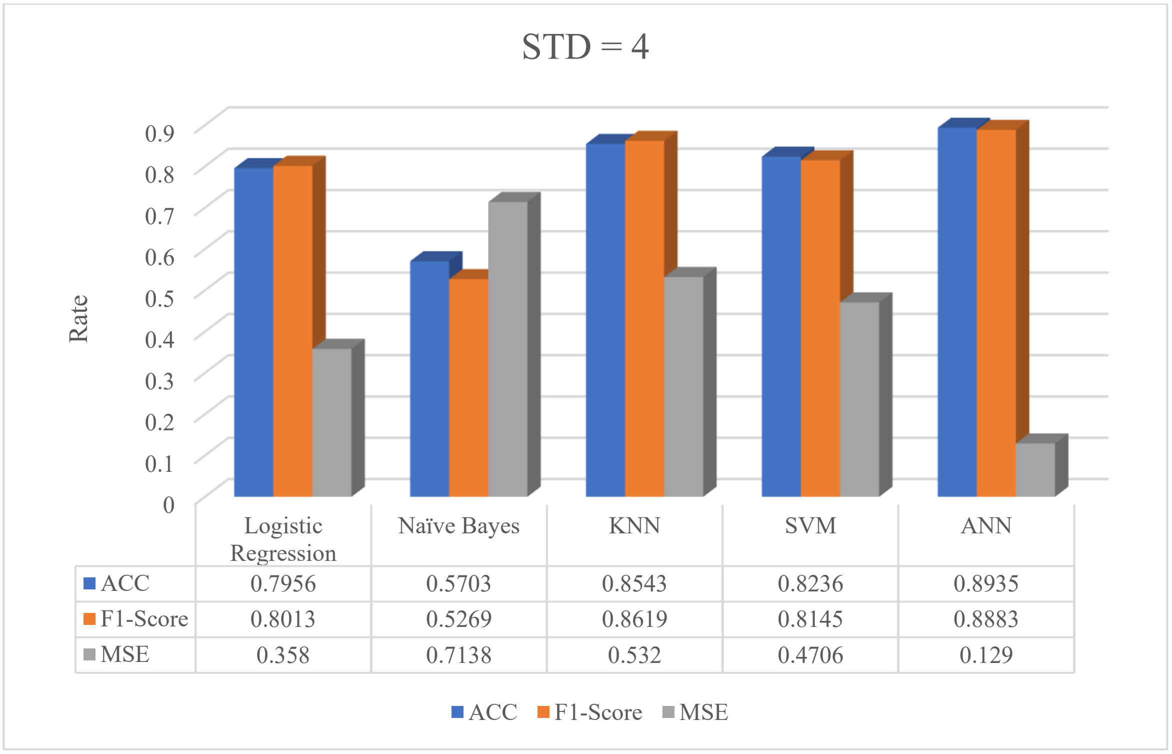
	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
■ ACC	0.9212	0.5798	0.9412	0.9031	0.9648
■ F1-Score	0.8759	0.5523	0.9025	0.8956	0.9543
■ MSE	0.12	0.6012	0.388	0.2508	0.0809

Figure 24 Gaussian correlated noise with std of 1 for the second dataset



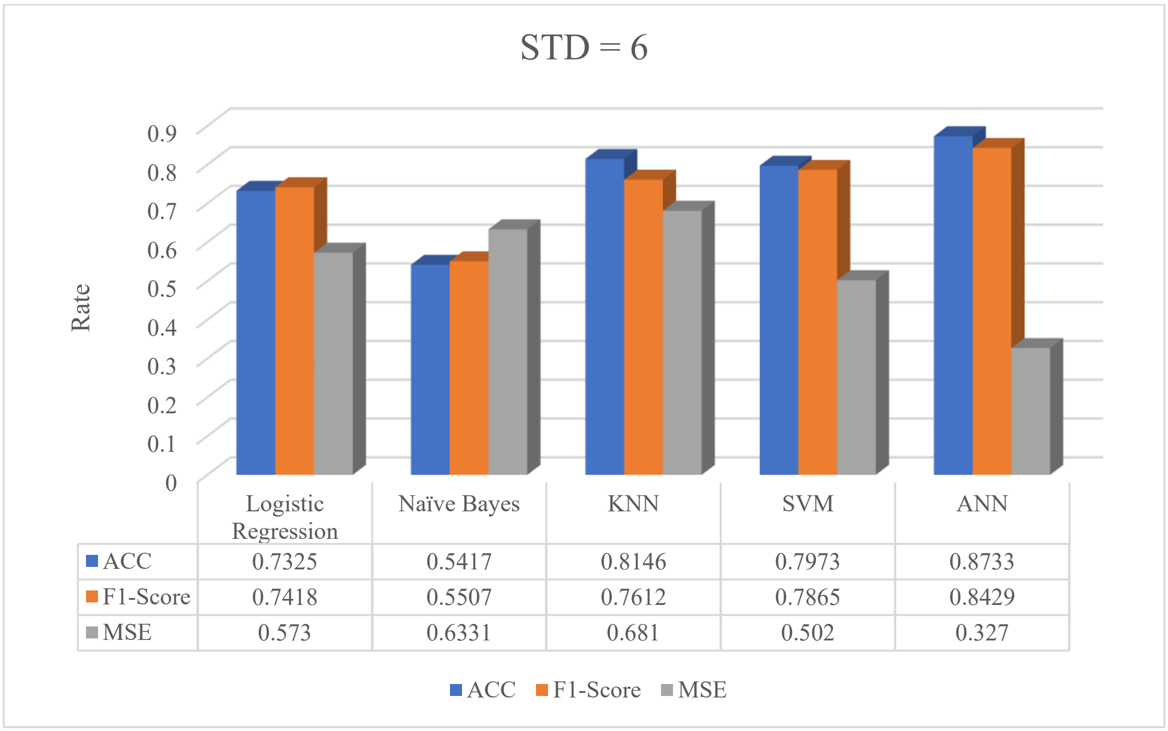
	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
■ ACC	0.8875	0.5619	0.9036	0.8696	0.901
■ F1-Score	0.8426	0.5369	0.8669	0.821	0.9145
■ MSE	0.239	0.6212	0.451	0.35	0.0958

Figure 25 Gaussian correlated noise with std of 2 for the second dataset



	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
■ ACC	0.7956	0.5703	0.8543	0.8236	0.8935
■ F1-Score	0.8013	0.5269	0.8619	0.8145	0.8883
■ MSE	0.358	0.7138	0.532	0.4706	0.129

Figure 26 Gaussian correlated noise with std of 4 for the second dataset



	Logistic Regression	Naïve Bayes	KNN	SVM	ANN
■ ACC	0.7325	0.5417	0.8146	0.7973	0.8733
■ F1-Score	0.7418	0.5507	0.7612	0.7865	0.8429
■ MSE	0.573	0.6331	0.681	0.502	0.327

Figure 27 Gaussian correlated noise with std of 6 for the second dataset

The results depicted in Figure 16 - 27 compare five distinct machine-learning techniques. Among them, ANN is the most capable of handling different correlated noises. It is worth noting that each noise used in this comparison has a varying standard deviation. This study investigates the system's robustness against different abnormalities that may occur in the real world. This is important not only for the system but also for its owner. To provide a more comprehensive analysis of the results, Figure 28 illustrates another form of comparison. The figure confirms that ANN outperforms the other techniques in handling the different types of correlated noises. The next two diagrams, Figure 28 and Figure 29, also focus on the main evaluation method considered in this work to illustrate comparing the five methods' accuracy in two different datasets.

In this work, an observation has emerged regarding the remarkable degree of proximity between the F1 Score and ACC metrics. This intriguing alignment between the two metrics indicates the model's robust performance and equilibrium.

The convergence of F1 Score and Accuracy metrics signal the model's consistent predictive prowess. The F1 Score captures the delicate balance between precision and recall, providing an insightful evaluation of the model's ability to accurately identify positive instances while achieving comprehensive coverage of positive cases. On the other hand, Accuracy quantifies the overall proportion of accurate predictions across all classes, reflecting the model's general predictive success.

This convergence, however, goes beyond mere coincidence. There are several compelling factors contributing to this phenomenon:

Balanced Equilibrium: The close correspondence between the F1 Score and Accuracy metrics signifies that this model maintains a harmonious equilibrium between minimizing false positives and negatives while ensuring accurate classification across instances. This equilibrium is a testament to the model's adeptness at delivering precise positive predictions and effectively classifying instances with accuracy on a broader scale.

Class Imbalance Mastery: The alignment suggests this model excels at handling class imbalance scenarios. Given the sensitivity of the F1 Score to the performance of the minority class, the proximity between the two metrics implies that this model successfully navigates this challenge, which is critical for real-world applications.

Calibration and Generalization: The model's well-calibrated predictions and consistent performance on training and validation datasets contribute to the convergence. This reflects the model's strong generalization, where it accurately captures the underlying patterns of the data and makes reliable predictions on unseen instances.

Feature Quality and Algorithm Suitability: The quality and relevance of features used for training, coupled with the suitability of chosen algorithm and hyperparameters, are key contributors. The alignment showcases that the model effectively leverages these factors to balance precision and recall.

Stability across Thresholds: This model's consistent performance across a range of threshold values for classification indicates its robustness and stability. This adaptability is crucial for different decision scenarios and ensures the model's reliability.

Confidence and Interpretability: The proximity between the two metrics enhances confidence in the model's performance. Furthermore, understanding the trade-offs between precision, recall, and overall accuracy allows one to comprehensively interpret the model's behaviour and decision, recall, and comprehensive classification proficiency.

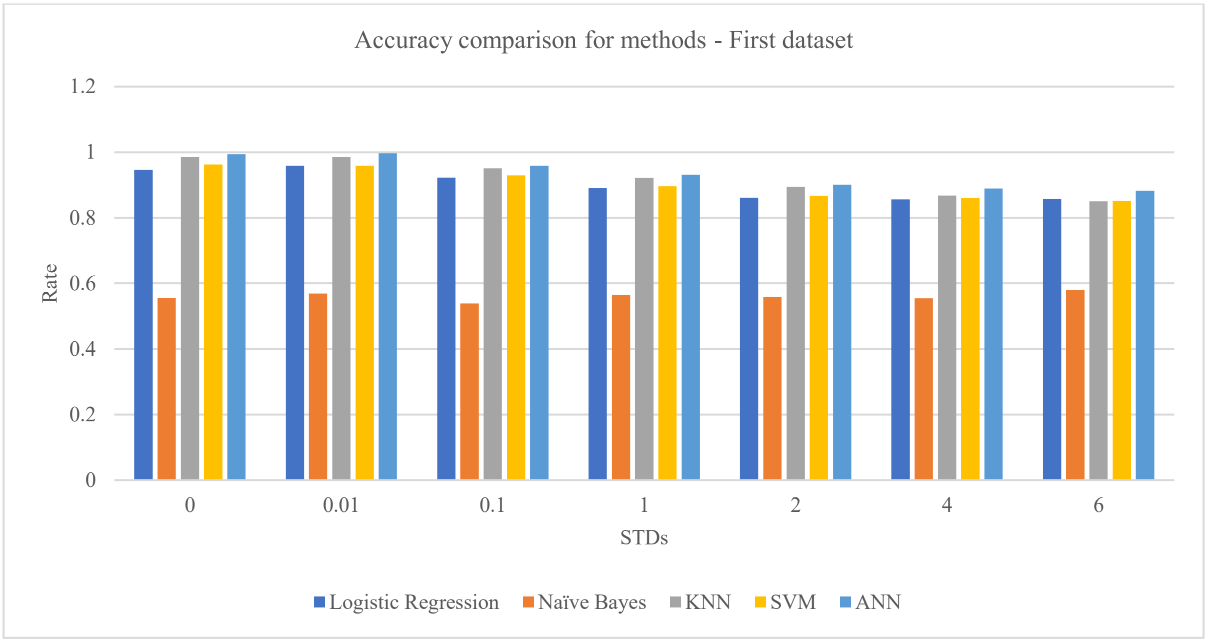


Figure 28 Accuracy, f1 Score, and mse of different methods with different noises strengths (first dataset)

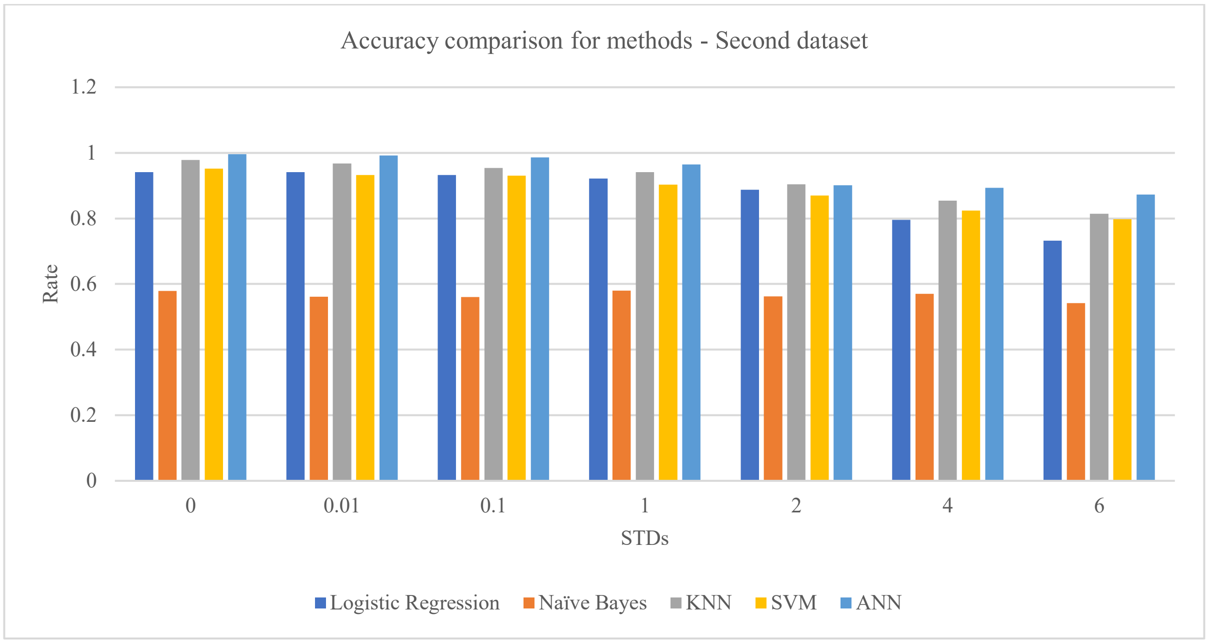


Figure 29 Accuracy, f1 Score, and mse of different methods with different noises strengths (second dataset)

In this section of the thesis, a meticulous comparison has been carried out between the predicted outcomes and the actual observed values generated by various machine learning algorithms. The primary focus of this comparison was to assess the performance of distinct algorithms – including ANN, Logistic Regression, Naive Bayes, KNN, and SVM – in predicting outcomes within a specific one-hour timeframe.

The presented figures within this section offer a visually insightful perspective, portraying the predicted values alongside the corresponding actual values for each algorithm. These figures illustrate the degree of conformity or divergence between the predicted and actual values for the designated time interval.

The comprehensive evaluation of these algorithms' predictive capabilities takes into account the respective accuracy levels, which are as follows: ANN exhibits an accuracy rate of 99%, while Logistic Regression displays an accuracy of 94%. Naive Bayes demonstrates an accuracy of 57%, and KNN showcases a high accuracy of 97%. Lastly, SVM with an accuracy of 95%.

It is important to note that this comparison is conducted within a controlled context, where the data has not yet been subjected to noise addition or correlation time consideration. This controlled environment allows a clear assessment of the algorithms' inherent predictive abilities, unobscured by external factors.

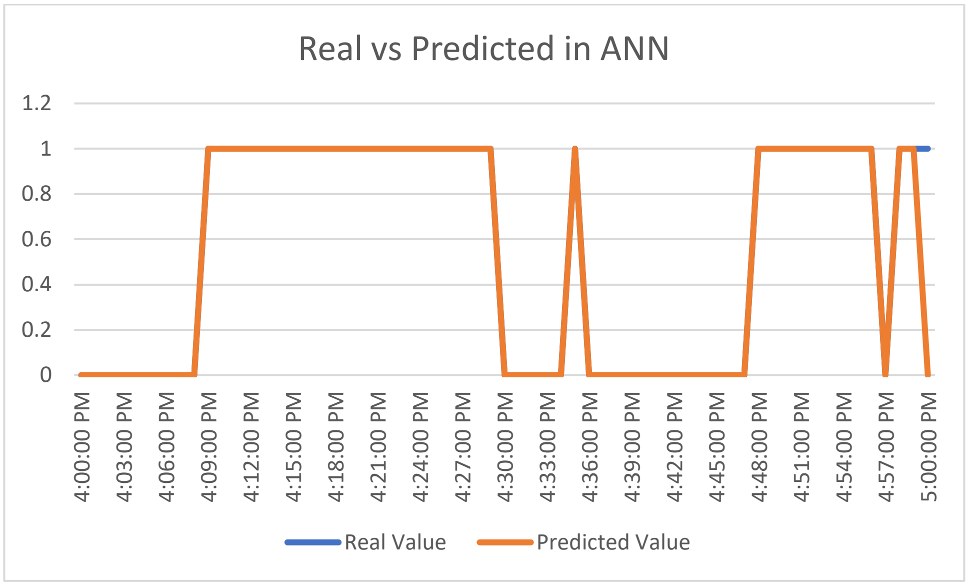


Figure 30 Real vs predicted in ANN

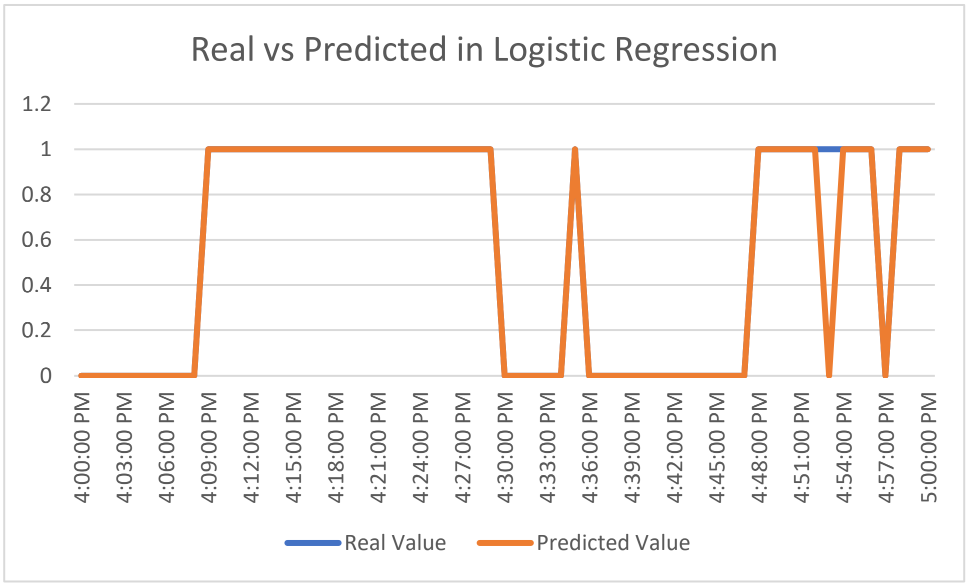


Figure 31 Real vs predicted in logistic regression

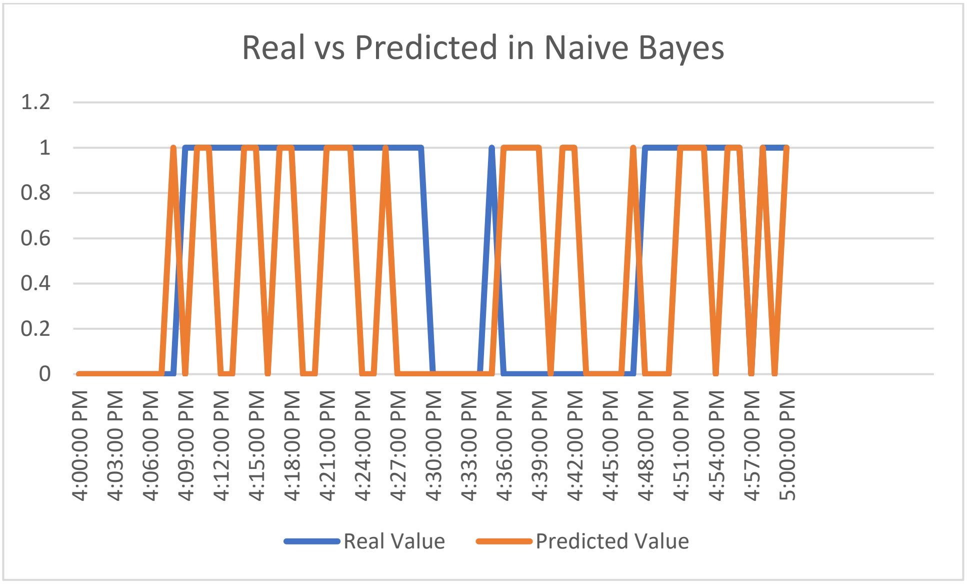


Figure 32 Real vs predicted in naive bayes

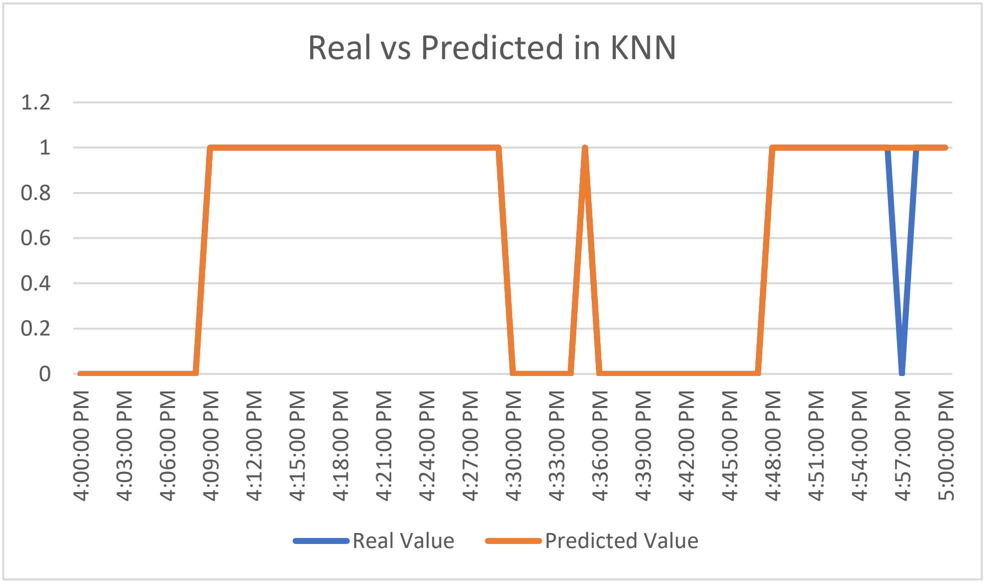


Figure 33 Real vs predicted in KNN

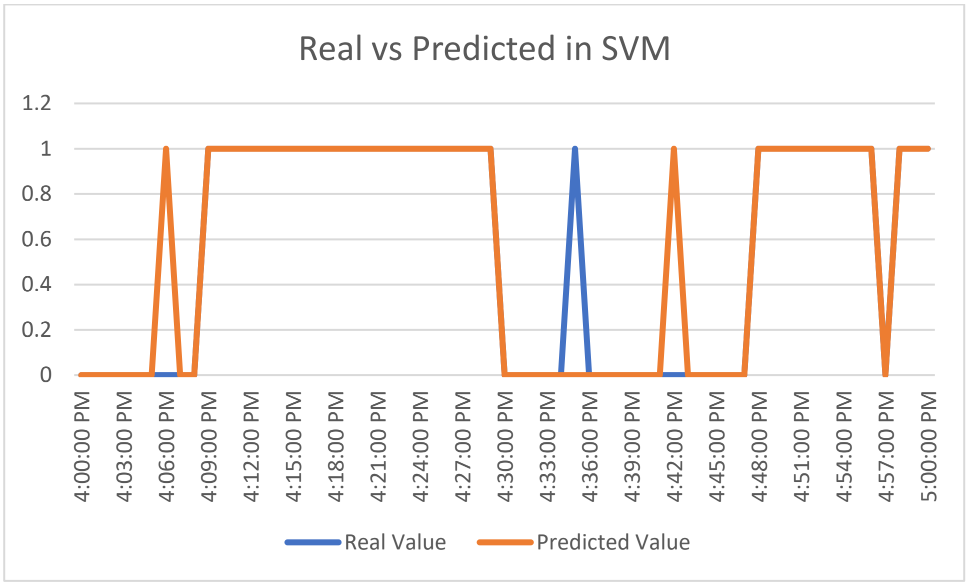


Figure 34 Real vs predicted in SVM

The decision to opt for six specific features over a larger set of ten was motivated by improving the model's accuracy and practicality, particularly during passive mode. Striking a balance between data complexity, computational efficiency, and interpretability was crucial, given the model's restrained actions in passive mode. The selected features—Owner choice, Battery SOC min, Time, Battery power consumption, Battery SOC, Grid power consumption, and Date—were carefully chosen due to their direct impact on the state of EV batteries. Avoiding additional features was a conscious choice to prevent potential confusion and overfitting, especially when the model's engagement is passive.

The "Owner choice" feature remains pivotal, reflecting varying user participation. Including "Battery SOC Min" for non-participating users lacks relevance, particularly when data is limited during passive periods. Similarly, "Power consumption min" and "Power consumption max" are already intrinsic to the system, obviating the need for the model to learn them even in passive mode. The model's accuracy and ability to offer accurate predictions, even during passive mode, validate the focus on these specific features. These features encompass fundamental aspects such as battery charge, temporal patterns, and grid interactions—critical for reliable predictions about battery states in passive scenarios.

Considering the passive mode's limitations, the goal remains to capture real-world scenarios without excessive complexity. The chosen attributes provide essential information for accurate predictions, ensuring the model's efficacy across varying conditions and its value in passive situations. Future considerations may introduce more factors, like weather conditions, for model enhancement. The model's performance meets expectations based on evaluation metrics, utilizing existing features for accurate predictions during passive phases.

Chapter 6 : Conclusion And Future Works

6.1 Conclusion

This study conducts comprehensive experiments using five prominent machine learning techniques: ANN, KNN, Naive Bayes, logistic regression, and SVM. The primary objective of this research is to identify the most accurate method among these approaches for a specific task. The evaluation criterion employed is accuracy, which measures the correctness of predictions.

A feature selection technique that involved analyzing the correlation matrix and determining feature importance is employed to ensure the robustness and effectiveness of the proposed models. The goal is to identify the most relevant input variables for achieving accurate predictions. When the raw data is applied, it is observed that the feature selection process does not significantly influence the accuracy of the models. Different trials are applied to check the importance of the feature selections and to see if it is needed. Three different correlation timescales (short, medium, and long) are used to validate the effectiveness of feature selection for the proposed models. Consequently, two features, namely Date and Battery Power Consumption, are identified as unimportant for the proposed models. Eliminating these features led to improving the accuracy and efficiency of the system.

Gaussian-correlated noise with varying strengths is introduced to assess the robustness of each technique in the presence of noise. Remarkably, even under noisy conditions, the Artificial Neural Network consistently demonstrated superior performance compared to the other techniques, reaffirming its accuracy and noise tolerance dominance.

It is worth noting that the data used in this study is obtained from the Gridlab software. Gridlab is a widely recognized software tool for simulating power distribution systems and analyzing their performance. By leveraging Gridlab, reliable and realistic data was obtained, enabling thorough experiments and meaningful conclusions.

6.2 Future Works

While this research has provided valuable insights into the performance of different machine learning techniques for the given task, there are several avenues for future exploration and improvement. The following areas offer potential directions for future work:

- **Refining Feature Selection Techniques**: The feature selection process significantly impacted accuracy, particularly by incorporating correlation time. Exploring additional feature selection techniques, such as recursive feature elimination, genetic algorithms, or embedded methods, could improve model performance. Additionally, investigating the impact of different correlation times and considering other feature relevance metrics may provide deeper insights into the most influential features for accurate predictions.
- **Handling Noisy Data**: Although the models exhibited strong performance in the presence of Gaussian-correlated noise, further investigation into the effects of different types and levels of noise would be valuable. Exploring noise reduction techniques, such as denoising algorithms or data augmentation approaches, can enhance the models' robustness and ability to handle real-world data.
- **Incorporating Temporal Information**: Power distribution systems often exhibit temporal dependencies, where past observations influence the current state. Integrating temporal information into the models, such as time series analysis techniques or recurrent neural networks, could capture these dependencies and potentially lead to improved predictions and decision-making.
- **Evaluating Energy Efficiency**: While accuracy is a crucial metric, evaluating the energy efficiency of the models can provide additional insights. Assessing different techniques' energy consumption and computational requirements can help identify approaches that deliver accurate predictions and optimize resource utilization.

- **Exploring Transfer Learning**: Investigating the application of transfer learning techniques can be beneficial, especially when dealing with limited labelled data. Leveraging knowledge from pre-trained models or related domains may enable the development of more effective models with improved generalization capabilities.
- **Real-world Deployment and Validation**: Validating the models in real-world power distribution systems is essential to ensure their practical viability. Collaborating with industry partners or power grid operators to conduct field trials or deploy the models in pilot projects can provide valuable feedback and insights into their performance, scalability, and applicability to real-world scenarios.

References

[1] EPA, “Carbon Pollution from Transportation,” Carbon Pollution from Transportation, May 19, 2021. https://www.epa.gov/transportation-air-pollution-and-climate-change/carbon-pollution-transportation (accessed Aug. 06, 2022).

[2] N. Garwa and K. R. Niazi, “‘Impact of EV on Integration with Grid System – A Review,’” in 2019 8th International Conference on Power Systems (ICPS), Jaipur, India: IEEE, Dec. 2019, pp. 1–6. doi: 10.1109/ICPS48983.2019.9067587.

[3] S. Morsalin, K. Mahmud, and G. Town, “Electric vehicle charge scheduling using an artificial neural network,” in 2016 IEEE Innovative Smart Grid Technologies - Asia (ISGT-Asia), Melbourne, Australia: IEEE, Nov. 2016, pp. 276–280. doi: 10.1109/ISGT-Asia.2016.7796398.

[4] G. M. Asim Akhtar, A. T. Al-Awami, E. Sortomme, M. A. Abido, and M. W. Ahmed, “Autonomous electric vehicle charging management over real time digital simulator,” in 2014 IEEE PES General Meeting | Conference & Exposition, National Harbor, MD: IEEE, Jul. 2014, pp. 1–5. doi: 10.1109/PESGM.2014.6939355.

[5] F. M. Shakeel and O. P. Malik, “Vehicle-To-Grid Technology in a Micro-grid Using DC Fast Charging Architecture,” in 2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE), Edmonton, AB, Canada: IEEE, May 2019, pp. 1–4. doi: 10.1109/CCECE.2019.8861592.

[6] A. Arancibia and K. Strunz, “Modeling of an electric vehicle charging station for fast DC charging,” in 2012 IEEE International Electric Vehicle Conference, Greenville, SC, USA: IEEE, Mar. 2012, pp. 1–6. doi: 10.1109/IEVC.2012.6183232.

[7] G. B. Sahinler and G. Poyrazoglu, “Modeling and Simulation of Three-Phase Bidirectional Electric Vehicle Charger,” in 2021 3rd Global Power, Energy and Communication Conference (GPECOM), Antalya, Turkey: IEEE, Oct. 2021, pp. 21–27. doi: 10.1109/GPECOM52585.2021.9587631.

[8] E. H. E. Bayoumi, “Design of Three-Phase LCL-Filter for Grid-Connected PWM Voltage Source Inverter Using Bacteria Foraging Optimization,” in Electricity Distribution, P. Karampelas and L. Ekonomou, Eds., in Energy Systems. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 199–219. doi: 10.1007/978-3-662-49434-9_8.

[9] S. R. B. and V. C., “Analysis of modified plug-in electric vehicle charger controller with grid support functionalities,” PLoS ONE, vol. 17, no. 1, p. e0262365, Jan. 2022, doi: 10.1371/journal.pone.0262365.

[10] P. K. Mohanty, P. Jena, and N. P. Padhy, “Home Electric Vehicle Charge Scheduling Using Machine Learning Technique,” in 2020 IEEE International Conference on Power Systems Technology (POWERCON), Bangalore, India: IEEE, Sep. 2020, pp. 1–5. doi: 10.1109/POWERCON48463.2020.9230627.

[11] N. Mhaisen, N. Fetais, and A. Massoud, “Real-Time Scheduling for Electric Vehicles Charging/Discharging Using Reinforcement Learning,” in 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT), Doha, Qatar: IEEE, Feb. 2020, pp. 1–6. doi: 10.1109/ICIoT48696.2020.9089471.

[12] D. A. Renata et al., “Modeling of Electric Vehicle Charging Energy Consumption using Machine Learning,” in 2021 International Conference on Advanced Computer Science and Information Systems (ICACSIS), Depok, Indonesia: IEEE, Oct. 2021, pp. 1–6. doi: 10.1109/ICACSIS53237.2021.9631324.

[13] A. Thingvad, L. Calearo, P. B. Andersen, and M. Marinelli, “Empirical Capacity Measurements of Electric Vehicles Subject to Battery Degradation From V2G Services,” IEEE Trans. Veh. Technol., vol. 70, no. 8, pp. 7547–7557, Aug. 2021, doi: 10.1109/TVT.2021.3093161.

[14] “Form EIA-861M (formerly EIA-826) detailed data.” https://www.eia.gov/electricity/data/eia861m/#netmeter

[15] Q. Dang, D. Wu, and B. Boulet, “Electric Vehicle Battery as Energy Storage Unit Consider Renewable Power Uncertainty,” in 2021 IEEE Energy Conversion Congress and Exposition (ECCE), Vancouver, BC, Canada: IEEE, Oct. 2021, pp. 668–673. doi: 10.1109/ECCE47101.2021.9595375.

[16] M. H. S. M. Haram, J. W. Lee, G. Ramasamy, E. E. Ngu, S. P. Thiagarajah, and Y. H. Lee, “Feasibility of utilising second life EV batteries: Applications, lifespan, economics, environmental impact, assessment, and challenges,” Alexandria Engineering Journal, vol. 60, no. 5, pp. 4517–4536, Oct. 2021, doi: 10.1016/j.aej.2021.03.021.

[17] S. Alshahrani, M. Khalid, and M. Almuhaini, “Electric Vehicles Beyond Energy Storage and Modern Power Networks: Challenges and Applications,” IEEE Access, vol. 7, pp. 99031–99064, 2019, doi: 10.1109/ACCESS.2019.2928639.

[18] X. Yan, B. Guan, and X. Du, “Bidirectional Charging Strategy of Electric Vehicle based on Predictive Control Method,” in 2021 International Conference on Power System Technology (POWERCON), Haikou, China: IEEE, Dec. 2021, pp. 789–793. doi: 10.1109/POWERCON53785.2021.9697595.

[19] G. Salvatti, E. Carati, R. Cardoso, J. da Costa, and C. Stein, “Electric Vehicles Energy Management with V2G/G2V Multifactor Optimization of Smart Grids,” Energies, vol. 13, no. 5, p. 1191, Mar. 2020, doi: 10.3390/en13051191.

[20] Jianwei Chen, Yuhang Wang, Guibin Wang, Zhao Xu, and Chaohang Zhang, “Feasibility analysis and comparison of different types of electric vehicles,” in 10th International Conference on Advances in Power System Control, Operation & Management (APSCOM 2015), Hong Kong, China: Institution of Engineering and Technology, 2015, p. 3 (6 .)-3 (6 .). doi: 10.1049/ic.2015.0215.

[21] V. Totev, V. Gueorgiev, and P. Rizov, “Regenerative braking of electric vehicles,” in 2019 11th Electrical Engineering Faculty Conference (BulEF), Varna, Bulgaria: IEEE, Sep. 2019, pp. 1–5. doi: 10.1109/BulEF48056.2019.9030768.

[22] M. Safayatullah, M. T. Elrais, S. Ghosh, R. Rezaii, and I. Batarseh, “A Comprehensive Review of Power Converter Topologies and Control Methods for Electric Vehicle Fast Charging Applications,” IEEE Access, vol. 10, pp. 40753–40793, 2022, doi: 10.1109/ACCESS.2022.3166935.

[23] V.-B. Vu, J. M. Gonzalez-Gonzalez, V. Pickert, M. Dahidah, and A. Trivino, “A Hybrid Charger of Conductive and Inductive Modes for Electric Vehicles,” IEEE Trans. Ind. Electron., vol. 68, no. 12, pp. 12021–12033, Dec. 2021, doi: 10.1109/TIE.2020.3042162.

[24] S. Habib, M. M. Khan, F. Abbas, and H. Tang, “Assessment of electric vehicles concerning impacts, charging infrastructure with unidirectional and bidirectional chargers, and power flow comparisons,” Int J Energy Res, vol. 42, no. 11, pp. 3416–3441, Sep. 2018, doi: 10.1002/er.4033.

[25] G. Sree lakshmi, G. Divya, and G. Sravani, “V2G Transfer of Energy to Various Applications,” E3S Web Conf., vol. 87, p. 01019, 2019, doi: 10.1051/e3sconf/20198701019.

[26] Hamed H. H. Aly “A hybrid optimized model of Adaptive Neuro-Fuzzy Inference System, Recurrent Kalman Filter and Neuro-Wavelet for Wind Power Forecasting Driven by DFIG”, Volume 239, Part E, Energy 2022, https://doi.org/10.1016/j.energy.2021.122367.

[27] X. Yan, B. Zhang, X. Xiao, H. Zhao, and L. Yang, “A bidirectional power converter for electric vehicles in V2G systems,” in 2013 International Electric Machines & Drives Conference, Chicago, IL, USA: IEEE, May 2013, pp. 254–259. doi: 10.1109/IEMDC.2013.6556261.

[28] U. C. Chukwu, “The Impact of V2G on the Distribution System: Power Factors and Power Loss Issues,” in 2019 SoutheastCon, Huntsville, AL, USA: IEEE, Apr. 2019, pp. 1–4. doi: 10.1109/SoutheastCon42311.2019.9020481.

[29] B. Sturmberg, “Owners of Electric Vehicles to be Paid to Plug into the Grid to Help Avoid Blackouts,” Jul. 08, 2020. https://www.gizmodo.com.au/2020/07/owners-of-electric-vehicles-to-be-paid-to-plug-into-the-grid-to-help-avoid-blackouts/ (accessed Jun. 08, 2022).

[30] G. Bhutada, “Breaking Down the Cost of an EV Battery Cell,” Feb. 2022. https://www.visualcapitalist.com/breaking-down-the-cost-of-an-ev-battery-cell/ (accessed Jun. 08, 2022).

[31] G. Lacey, G. Putrus, and E. Bentley, “Smart EV charging schedules: supporting the grid and protecting battery life,” IET Electrical Systems in Transportation, vol. 7, no. 1, pp. 84–91, Mar. 2017, doi: 10.1049/iet-est.2016.0032.

[32] Hmeda Musbah, Gama Ali, Hamed H Aly, Timothy A Little " Energy management using multi-criteria decision making and machine learning classification algorithms for intelligent system" Electric Power Systems Research, vol 203, 2022, https://doi.org/10.1016/j.epsr.2021.107645.

[33] Hamed H. Aly “A Novel Approach for Tidal Currents Harmonic Constitutions Forecasting Hybrid Models based on Clustering Techniques for Smart Grid” International Journal of Renewable Energy, Volume 147, Part 1, March 2020, Pages 1554-1564.

[34] E. Sortomme and M. A. El-Sharkawi, “Optimal Combined Bidding of Vehicle-to-Grid Ancillary Services,” IEEE Trans. Smart Grid, vol. 3, no. 1, pp. 70–79, Mar. 2012, doi: 10.1109/TSG.2011.2170099.

[35] K. M. Tan, D. V. K. Ramachandaramurthy, and M. J. Y. Yong, “Bidirectional Battery Charger for Electric Vehicle,” p. 6, 2014.

[36] Ke Bao, Shuhui Li, and Huiying Zheng, “Battery charge and discharge control for energy management in EV and utility integration,” in 2012 IEEE Power and Energy Society General Meeting, San Diego, CA: IEEE, Jul. 2012, pp. 1–8. doi: 10.1109/PESGM.2012.6344719.

[37] Z. Wang and S. Wang, “Grid Power Peak Shaving and Valley Filling Using Vehicle-to-Grid Systems,” IEEE Trans. Power Delivery, vol. 28, no. 3, pp. 1822–1829, Jul. 2013, doi: 10.1109/TPWRD.2013.2264497.

[38] J. Guo, J. Yang, Z. Lin, C. Serrano, and A. M. Cortes, “Impact Analysis of V2G Services on EV Battery Degradation -A Review,” in 2019 IEEE Milan PowerTech, Milan, Italy: IEEE, Jun. 2019, pp. 1–6. doi: 10.1109/PTC.2019.8810982.

[39] D. B. Richardson, “Electric vehicles and the electric grid: A review of modelling approaches, Impacts, and renewable energy integration,” Renewable and Sustainable Energy Reviews, vol. 19, pp. 247–254, Mar. 2013, doi: 10.1016/j.rser.2012.11.042.

[40] Q. Zhang, C. Li, and Y. Wu, “Analysis of Research and Development Trend of the Battery Technology in Electric Vehicle with the Perspective of Patent,” Energy Procedia, vol. 105, pp. 4274–4280, May 2017, doi: 10.1016/j.egypro.2017.03.918.

[41] J. Jenkins, “High-voltage EV battery packs: benefits and challenges. More voltage, more better?,” 2021. https://chargedevs.com/features/high-voltage-ev-battery-packs-benefits-and-challenges-more-voltage-more-better/

[42] A. Millner, “Modeling Lithium Ion battery degradation in electric vehicles,” in 2010 IEEE Conference on Innovative Technologies for an Efficient and Reliable Electricity Supply, Waltham, MA, USA: IEEE, Sep. 2010, pp. 349–356. doi: 10.1109/CITRES.2010.5619782.

[43] O. Tremblay, L.-A. Dessaint, and A.-I. Dekkiche, “A Generic Battery Model for the Dynamic Simulation of Hybrid Electric Vehicles,” in 2007 IEEE Vehicle Power and Propulsion Conference, Arlington, TX, USA: IEEE, Sep. 2007, pp. 284–289. doi: 10.1109/VPPC.2007.4544139.

[44] D. D.L. Chung, Carbon Composites. 2017.

[45] S. Li, T. A. Haskew, Y.-K. Hong, and L. Xu, “Direct-current vector control of three-phase grid-connected rectifier–inverter,” Electric Power Systems Research, vol. 81, no. 2, pp. 357–366, Feb. 2011, doi: 10.1016/j.epsr.2010.09.011.

[46] Hamed H. H. Aly" A proposed intelligent short-term load forecasting hybrid models of ANN, WNN and KF based on clustering techniques for smart grid" Journal of Electric Power Systems Research, Vol 182, May 2020.

[47] Seldon, "Supervised vs Unsupervised Learning Explained," Oct. 16, 2021. https://www.seldon.io/supervised-vs-unsupervised-learning-explained#:~:text=The%20main%20difference%20between%20supervised,processes%20unlabelled%20or%20raw%20data. (accessed Aug. 21, 2022).

[48] V. Kanade, "What Is Machine Learning? Definition, Types, Applications, and Trends for 2022," spiceworks. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-ml/

[49] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. in Adaptive computation and machine learning. Cambridge, Massachusetts: The MIT Press, 2016.

[50] N. Aboueata, S. Alrasbi, A. Erbad, A. Kassler, and D. Bhamare, "Supervised Machine Learning Techniques for Efficient Network Intrusion Detection," in 2019 28th International Conference on Computer Communication and Networks (ICCCN), Valencia, Spain: IEEE, Jul. 2019, pp. 1–8. doi: 10.1109/ICCCN.2019.8847179.

[51] Hamed Aly, "Forecasting, Modeling and Control of Tidal Currents Electrical Energy Systems" Ph.D. dissertation, Department of Electrical and Computer Engineering, Dalhousie University, Halifax, N.S., 2012.

[52] A. R. Hadd and J. L. Rodgers, Understanding correlation matrices, First Edition. in Quantitative applications in the social sciences, no. 186. Los Angeles: SAGE Publications, Inc, 2021.

[53] Y. Zhang, H. Zhang, and B. Zhang, "An Effective Ensemble Automatic Feature Selection Method for Network Intrusion Detection," Information, vol. 13, no. 7, p. 314, Jun. 2022, doi: 10.3390/info13070314.

[54] B. Charbuty and A. Abdulazeez, “Classification Based on Decision Tree Algorithm for Machine Learning,” JASTT, vol. 2, no. 01, pp. 20–28, Mar. 2021, doi: 10.38094/jastt20165.

[55] S. Chauhan and S. Lee, “Machine Learning-Based Anomaly Detection for Multivariate Time Series With Correlation Dependency,” IEEE Access, vol. 10, pp. 132062–132070, 2022, doi: 10.1109/ACCESS.2022.3230352.

[56] H. A. Bedel, I. Sivgin, and T. Cukur, “A Graphical Network Layer for Lagged Analysis of FMRI Data,” in 2022 30th Signal Processing and Communications Applications Conference (SIU), Safranbolu, Turkey: IEEE, May 2022, pp. 1–4. doi: 10.1109/SIU55565.2022.9864826.

[57] Y. Makinen, L. Azzari, and A. Foi, “Collaborative Filtering of Correlated Noise: Exact Transform-Domain Variance for Improved Shrinkage and Patch Matching,” IEEE Trans. on Image Process., vol. 29, pp. 8339–8354, 2020, doi: 10.1109/TIP.2020.3014721.

[58] “A Simple Technique for the Generation of Correlated Random Number Sequences,” IEEE Trans. Syst., Man, Cybern., vol. 9, no. 2, pp. 96–102, 1979, doi: 10.1109/TSMC.1979.4310156.

[59] “GridLab.” https://gridlab.org/

[60] P. Gotseff and B. Lundstrom, “Data-Driven Residential Load Modeling and Validation in GridLAB-D,” in 2017 Ninth Annual IEEE Green Technologies Conference (GreenTech), Denver, CO, USA: IEEE, Mar. 2017, pp. 20–25. doi: 10.1109/GreenTech.2017.9.

[61] G. Allen et al., “The GridLab grid application toolkit,” in Proceedings 11th IEEE International Symposium on High Performance Distributed Computing, Edinburgh, UK: IEEE Comput. Soc, 2002, p. 411. doi: 10.1109/HPDC.2002.1029941.

[62] “GridLab Publications.” https://gridlab.org/publications/

[63] K. Kaggle, “Household Electric Power Consumption,” 2006. https://www.kaggle.com/datasets/uciml/electric-power-consumption-data-set?resource=download (accessed Aug. 01, 2022).

[64] K. M. Kahloot and P. Ekler, “Algorithmic Splitting: A Method for Dataset Preparation,” IEEE Access, vol. 9, pp. 125229–125237, 2021, doi: 10.1109/ACCESS.2021.3110745.

[65] NumPy, “What is NumPy?,” Jun. 27, 2022. https://numpy.org/doc/stable/user/whatisnumpy.html

[66] ActiveState, “What Is Pandas In Python? Everything You Need To Know.” https://www.activestate.com/resources/quick-reads/what-is-pandas-in-python-everything-you-need-to-know/ (accessed Aug. 09, 2022).

[67] techopedia, “Scikit-Learn.” https://www.techopedia.com/definition/33860/scikit-learn#:~:text=Scikit%2Dlearn%20is%20a%20key,for%20many%20machine%20learning%20technologies. (accessed Sep. 02, 2022).

[68] Matplotlib, “Matplotlib: Visualization with Python.” https://matplotlib.org/ (accessed Aug. 30, 2012).

[69] Keras, “Keras.” https://keras.io/ (accessed Apr. 15, 2015).

[70] O. Surakhi et al., “Time-Lag Selection for Time-Series Forecasting Using Neural Network and Heuristic Algorithm,” Electronics, vol. 10, no. 20, p. 2518, Oct. 2021, doi: 10.3390/electronics10202518.

Appendix

Appendix A

Python Coding

```
from keras.models import Sequential
from keras.layers import Dense
import timeit
import numpy as np
import pandas as pd
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler
from statsmodels.tsa.arima.model import ARIMA
from sklearn.datasets import make_regression
from sklearn.feature_selection import RFECV
from sklearn.tree import DecisionTreeRegressor

dataset = pd.read_excel('Python Data 3.xlsx')
shuffled_df = dataset.sample(frac=1, random_state=2)
shuffled_df.head(7)
dataset2 = shuffled_df.sample(n=50000)
```

In this research section, the code snippet showcases the foundational libraries and techniques utilized to implement and analyse the predictive model. The code demonstrates the inclusion and utilization of essential libraries such as **Keras**, **NumPy**, **pandas**, and **matplotlib**, all integral to constructing the predictive model.

The script begins with importing the necessary components, such as the **Sequential** model and **Dense** layer from **Keras**, pivotal for constructing neural network architectures. Additionally, the inclusion of

timeit offers a mechanism for measuring the execution time of the implemented processes, contributing to the robust evaluation of the model's efficiency.

The script integrates functionalities from NumPy and pandas to prepare the data for modelling. The imported dataset, sourced from an Excel file **('Python Data 3.xlsx')**, is loaded into a **pandas DataFrame**. Furthermore, the code snippet incorporates data preprocessing techniques, including **shuffling** the dataset to mitigate inherent biases, as evidenced by utilising the **`sample()`** function with a specified random seed.

The subsequent code segment involves subsampling the shuffled dataset to **50,000** samples, denoted by the **`sample(n=50000)`** function call. This subset is meticulously crafted to be representative of the entire dataset, ensuring an accurate representation of data points within a controlled computational scope.

This code segment serves as the preliminary step toward the model construction process, laying the foundation for data manipulation and preparation, which is vital for generating meaningful insights and accurate predictions. The script's logical organization underscores the researcher's meticulous approach to data handling and model preparation, thus setting the stage for the subsequent phases of the research.

```
# Assuming 'dataset2' contains your data

x = pd.DataFrame(dataset2, columns=['Owner 2','<SOC (%)> 2','Grid
consumption','date','time','Power Battery 2'])
y = pd.DataFrame(dataset2, columns=['Charge/Discharge/Ideal Detection2'])
x_columns = len(x.columns)

# Introduce time lagged features for short correlation time
lagged_steps = 5
lagged_dfs = []

for lag in range(1, lagged_steps+1):
    lagged_cols = []
    for column in x:
        lagged_col = f'{column}_lag{lag}'
        lagged_cols.append(x[column].shift(lag))
    lagged_df = pd.concat(lagged_cols, axis=1)
    lagged_df.columns = [f'{col}_lag{lag}' for col in lagged_df.columns]
    lagged_dfs.append(lagged_df)

x = pd.concat([x] + lagged_dfs, axis=1).dropna()  # Concatenate lagged columns
with original DataFrame

y = y.loc[x.index]  # Drop corresponding rows from the target variable


# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3,
random_state=3)
```

This code excerpt showcases a systematic approach to data preprocessing and preparation, setting the stage for model training and evaluation. The steps taken underscore the researcher's commitment to ensuring the integrity and robustness of the predictive model.

The initial section of the code involves constructing two **DataFrames**, **'x'** and **'y'**, from the **'dataset2'**. **'x'** is thoughtfully assembled with specific columns **('Owner 2', '<SOC (%)> 2', 'Grid consumption', 'date', 'time', 'Power Battery 2')** chosen to serve as predictor variables. Meanwhile, **'y'** is structured with the **'Charge/Discharge/Ideal Detection2'** column, representing the target variable to be predicted. The careful selection of columns reflects the researcher's keen focus on variables integral to the predictive task.

The code then creates **time-lagged** features to capture short correlation patterns within the data. This is achieved by generating lagged versions of each column in **'x'** over a specified range of time steps (here, 1 to 5 lagged steps). This approach acknowledges the importance of temporal relationships in the data and endeavours to capture such dependencies through lagged features.

Through an iterative process, the lagged features are concatenated to the original **'x' DataFrame**, forming an enriched dataset that captures both the original variables and their lagged counterparts. The **'dropna()'** operation ensures that any rows with missing values resulting from the creation of lagged features are removed, preserving data integrity.

Furthermore, **'y'** is aligned with the revised **'x' DataFrame** by retaining only those rows that exist in both **DataFrames**. This synchronization is fundamental to guarantee the accuracy and coherence of the target variable with the predictor variables.

The code concludes by splitting the data into training and testing sets using the **'train_test_split'** function. The division is strategic, allocating **70%** of the data for training **('X_train', 'y_train')** and the remaining **30%** for testing **('X_test', 'y_test')**. The **'random_state'** parameter ensures reproducibility in the random selection process, contributing to the consistency of results.

This code snippet exemplifies the researcher's meticulous approach to preparing the dataset for model training, capturing temporal dependencies through lagged features, and strategically partitioning the data for training and testing. The logical sequence of operations demonstrates a comprehensive data preprocessing strategy to optimise the model's accuracy and predictive capabilities.

```
x = pd.DataFrame(dataset2, columns=['time','Owner 2','<SOC (%)> 2','Grid
consumption','date','Power Battery 2'])
y = pd.DataFrame(dataset2, columns=['Charge/Discharge/Ideal Detection1'])
x_columns = len(x.columns)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( x, y, test_size = 0.3,
random_state = 3)
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)


# Convert X_train numpy array to pandas dataframe
X_train = pd.DataFrame(X_train, columns=x.columns)

# Step 1: Create noise array
noise = np.random.randn(*X_train.shape)

# Step 2: Calculate correlation matrix
corr_matrix = np.corrcoef(X_train.T)

# Step 3: Get lower triangle of Cholesky decomposition
```

```
cholesky = np.linalg.cholesky(corr_matrix).T

# Step 4: Multiply noise by lower triangle and adjust standard deviation
std_dev = 6  # adjust this value to modify noise level
corr_noise = (std_dev * noise) @ cholesky

# Step 5: Add correlated noise to selected columns
X_train.loc[:, 'Grid consumption'] += corr_noise[:, 3]
```

The initial line of code converts the **X_train numpy** array into a Pandas **DataFrame**. **X_train** likely contains the training data, and the code assigns column names to the **DataFrame** using **x.columns**. This operation is helpful when you want to work with your data in a tabular format and utilize the functionalities provided by Pandas for data manipulation and analysis.

Next, a noise array is generated using **NumPy's np. random.randn()** function. This function produces random numbers sampled from a standard normal distribution (mean 0 and standard deviation 1). The ***X_train.shape** syntax unpacks the shape of the **X_train DataFrame**, which determines the dimensions of the generated noise array. This noise array is used to introduce randomness into the correlation structure of the data.

The code computes the correlation matrix of the transposed **X_train DataFrame** using **np.corrcoef(X_train.T)**. The .T transposition operator is applied to the DataFrame to ensure that rows become columns and columns become rows, as the **corrcoef** function expects variables to be in columns. The correlation matrix provides insights into the linear relationships between pairs of variables in the data.

The **Cholesky** decomposition is performed on the correlation matrix using **np.linalg.cholesky(corr_matrix)**. Cholesky decomposition is a technique that factorizes a symmetric positive definite matrix into the product of a lower triangular matrix and its transpose. In this case, the transpose of the Cholesky decomposition is taken using .T. The resulting lower triangular matrix captures the relationships between variables and helps create correlated noise.

A standard deviation value **std_dev** is set to control the level of noise added to the data. This value affects the spread or magnitude of the generated noise. Larger values of std_dev result in more pronounced noise, while smaller values lead to milder noise. Adjusting this value allows you to control the strength of the correlation-induced noise you'll add to the dataset.

Finally, correlated noise is added to a specific column in the **X_train DataFrame**. The line **X_train.loc[:, 'Grid consumption'] += corr_noise[:, 3]** adds the noise generated from the fourth column of **corr_noise** to the **'Grid consumption'** column of **X_train**. This step introduces correlated noise to the selected column, influenced by the correlations derived from the Cholesky decomposition. The chosen std_dev value scales the noise, so the larger the std_dev, the more impactful the noise will be on the data in that particular column.

Logistic Regression

```
from sklearn.linear_model import LogisticRegression
log_reg_model = LogisticRegression()
# Train (fit) the model
log_reg_model.fit(X_train, y_train)

# Make predictions
y_pred = log_reg_model.predict(X_test) # Predictions
y_true = y_test # True values

# Model evaluation
from sklearn.metrics import accuracy_score,mean_squared_error
from sklearn.metrics import precision_recall_fscore_support
import numpy as np

print("Accuracy:", np.round(accuracy_score(y_true, y_pred), 4))
print('MSE', mean_squared_error(y_test, y_pred))
from sklearn.metrics import classification_report
target_names = ['-1','0','1']
print(classification_report(y_test, y_pred, target_names=target_names))
# plt(y_test, y_pred)
# Make the confusion matrix

f1 = f1_score(y_test, y_pred, average='macro')
print('F1 score:', f1)
```

The code begins by importing the **LogisticRegression** class from the **sklearn.linear_model module**. This class is a part of the **Scikit-learn library**, which provides various tools for machine learning. An instance of the LogisticRegression model is created using the line **log_reg_model = LogisticRegression()**. This initializes a logistic regression model with default settings. Next, the model is trained or fitted using the training data with the **line log_reg_model.fit(X_train, y_train)**. The **X_train** variable represents the feature data, while **y_train** contains the corresponding labels or target values. The model learns from this data to make predictions later.

After training, the trained logistic regression model is used to make predictions on the test data. This is done using the line **y_pred = log_reg_model.predict(X_test)**. The predicted values are stored in the **y_pred** variable, while the actual true values are already available in the **y_test** variable.

The code calculates the **Accuracy** of the model's predictions using **accuracy_score(y_true, y_pred)**, where **y_true** contains the true labels, and **y_pred** contains the predicted labels. The accuracy score measures the proportion of correctly predicted instances among all instances. The **MSE** is computed using **mean_squared_error(y_test, y_pred)**. The **classification_report** function generates a detailed report with metrics like precision, recall, **F1-score**, and support for each class. The target class names are provided as **target_names**. The **F1 score** is calculated using **f1_score(y_test, y_pred, average='macro')**.

Naïve Bayes

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report, accuracy_score, f1_score,
mean_squared_error, confusion_matrix

# Train the model
nb = GaussianNB()
nb.fit(X_train, y_train)

# Test the model
```

```
y_pred = nb.predict(X_test)

# Evaluate the model
print(classification_report(y_test, y_pred))

accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred, average='weighted')
mse = mean_squared_error(y_test, y_pred)

# Print the evaluation metrics
print("Accuracy:", accuracy)
print("F1 Score:", f1)
print("Mean Squared Error:", mse)
```

The code starts by importing necessary libraries, **seaborn** for data visualization, **matplotlib.pyplot** for creating plots, **GaussianNB** from **sklearn.naive_bayes** for using the **Gaussian Naive Bayes classifier**, and various metrics from **sklearn.metrics** for evaluating the classifier's performance.

An instance of the Gaussian Naive Bayes classifier is created using **nb = GaussianNB()**. This classifier is a variant of the Naive Bayes algorithm that assumes the features are normally distributed within each class. Then, the classifier is trained on the training data using **nb.fit(X_train, y_train)**, where **X_train** contains the feature data and **y_train** contains the corresponding labels.

The trained model is tested on the test data using **y_pred = nb.predict(X_test)**. The predictions are stored in **y_pred**. The code proceeds to evaluate the model's performance using classification metrics. The **classification_report** function generates a detailed classification report containing metrics like precision, recall, F1-score, and support for each class. This report is printed using **print(classification_report(y_test, y_pred))**.

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, f1_score, mean_squared_error

# Initialize KNN classifier with k=5
knn = KNeighborsClassifier(n_neighbors=5)
```

```
# Train the model on the training data
knn.fit(X_train, y_train)

# Make predictions on the test data
y_pred = knn.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred, average='weighted')
mse = mean_squared_error(y_test, y_pred)
```

The code begins by importing necessary libraries: **KNeighborsClassifier** from **sklearn.neighbors** for using the K-Nearest Neighbors classifier, and various metrics from **sklearn.metrics** for evaluating the classifier's performance.

An instance of the K-Nearest Neighbors classifier is created using **knn = KNeighborsClassifier(n_neighbors=5)**. Here, **n_neighbors** is set to 5, which means the classifier will consider the 5 nearest neighbors when making predictions. KNN is a type of supervised machine learning algorithm used for classification tasks. This choice is often guided by the preference for odd numbers to avoid ties in classification, and five is a common starting point due to its balance between bias and variance. The selection of the number of neighbors should be based on the nature of the data, data density, and the trade-off between bias and variance. It's essential to experiment with different values, considering cross-validation, to find the optimal number of neighbors for your specific problem and dataset, as it can impact the model's generalization performance. Next, the KNN classifier is trained on the training data using **knn.fit(X_train, y_train)**, where **X_train** contains the feature data and **y_train** contains the corresponding labels. After training, the trained KNN classifier is used to make predictions on the **test data**. This is done using **y_pred = knn.predict(X_test)**. The predicted labels are stored in the **y_pred** variable.

SVM

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, f1_score, mean_squared_error

# Create an instance of SVM classifier
clf = SVC()

# Train the model on the preprocessed training data
clf.fit(X_train, y_train)

# Test the model on the preprocessed testing data
y_pred = clf.predict(X_test)

# Evaluate the model's accuracy
accuracy = clf.score(X_test, y_test)
print("Accuracy:", accuracy)
# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred, average='weighted')
mse = mean_squared_error(y_test, y_pred)
```

The code begins by importing necessary libraries: **SVC** from **sklearn.svm** for using the Support Vector Classifier, and various metrics from **sklearn.metrics** for evaluating the classifier's performance.

An instance of the Support Vector Classifier (SVC) is created using **clf = SVC**. SVC is a specialized instance of the SVM algorithm designed explicitly for classification tasks. By opting for SVC, the SVM framework aligned directly with the classification objectives, potentially leading to more streamlined implementation and interpretation of results for the particular project. Then, the model is trained on the preprocessed training data using **clf.fit(X_train, y_train)**, where **X_train** contains the feature data and **y_train** contains the corresponding labels.

The trained SVM classifier is tested on the preprocessed testing data using **y_pred = clf.predict(X_test)**. The predicted labels are stored in the **y_pred** variable. The model's accuracy is computed and printed using **accuracy = clf.score(X_test, y_test)**. The score method calculates the accuracy of the model's predictions using the test data **(X_test)** and true labels **(y_test)**. Further evaluation is also like the previous models.

```
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import mean_squared_error, accuracy_score, f1_score
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras. layers import Dense

# Assuming you have already defined X_train, X_test, y_train, and y_test
# Splitting data for training the Keras model
X_train_keras, X_val, y_train_keras, y_val = train_test_split(X_train,
y_train, test_size=0.2, random_state=42)

# Define the Keras ANN model
model_keras = Sequential()
model_keras.add(Dense(units=50, activation='relu',
input_dim=X_train.shape[1]))
model_keras.add(Dense(units=50, activation='relu'))
model_keras.add(Dense(units=num_classes, activation='softmax'))  # Change
num_classes to the number of output classes

# Compile the model
model_keras.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])

# Train the Keras model
history = model_keras.fit(X_train_keras, y_train_keras, epochs=10,
batch_size=32, validation_data=(X_val, y_val))

# Make predictions using the trained Keras model
y_pred_keras = model_keras.predict_classes(X_test)

# Calculate metrics
mse_keras = mean_squared_error(y_test, y_pred_keras)
acc_keras = accuracy_score(y_test, y_pred_keras)
f1_keras = f1_score(y_test, y_pred_keras, average='weighted')
```

The code begins by importing necessary libraries. Along with the libraries from the previous code, **train_test_split** is imported from **sklearn.model_selection** to split the training data into training and validation sets. **Sequential** and **Dense** are imported from **Keras**, which will be used to define the neural network architecture. The code assumes that **X_train**, **X_test**, **y_train**, and **y_test** are already defined. The training data is split into training and validation sets using train_test_split to use Keras. This is done to monitor the model's performance during training and prevent overfitting. The **test_size** parameter specifies the proportion of data to allocate for validation.

The Keras neural network model is defined using a **Sequential** model, representing a linear stack of layers. **Three Dense layers** are added to the model. The first two layers have 50 units each and use the **ReLU activation function**. The last layer has units equal to the number of output classes and uses the **softmax** activation function, which is suitable for multiclass classification problems.

The model is compiled using the compile method. The optimizer **'Adam'** is used, which is a popular optimization algorithm. The choice of using the Adam optimizer stems from its effectiveness in accelerating the convergence of neural network training. Adam, short for "Adaptive Moment Estimation," combines the benefits of two popular optimization algorithms, **AdaGrad** and **RMSProp**. Adam adapts the learning rates for each parameter in the model based on their historical gradients, allowing it to adjust learning rates dynamically and converge faster across varying dimensions and magnitudes of gradients. This adaptability is particularly valuable for deep learning tasks, where complex architectures may have layers with drastically different gradients. The ability of Adam to maintain learning rates per parameter contributes to the enhancement of training efficiency and the facilitation of quicker convergence towards optimal or near-optimal solutions. Considering these advantages, the Adam optimiser was chosen to expedite training and potentially attain improved outcomes in the specific neural network architecture. The loss function 'categorical_crossentropy' is appropriate for multiclass classification problems. The metric 'accuracy' is chosen to monitor the model's performance during training.

Appendix B

Gridlab Coding

```
clock {
    starttime '2023-08-15 00:00:00';
    stoptime '2023-08-15 01:00:00';
    timezone 'UTC';
}

module residential;

object house {
    parent electrical;
    name House1;
    heatgain_mode CONSTANT_LOAD;

    // Add other house-specific parameters

    ZIPload {
        name House1_Load;
        nominal_voltage 120;
        power_fraction_schedule Load_Profile_House1;
    }
}

object house {
    parent electrical;
    name House2;
    // Similar parameters and loads for House2
}

object house {
    parent electrical;
    name House3;
    // Similar parameters and loads for House3
}

object house {
    parent electrical;
    name House4;
    // Similar parameters and loads for House4
}

module ev;
object vehicle {
    parent electrical;
    name EV1;

    // Add EV-specific parameters

    ZIPload {
        name EV1_Load;
        nominal_voltage 120;
        power_fraction_schedule EV_Charging_Profile_EV1;
    }
}

object vehicle {
    parent electrical;
    name EV2;
    // Similar parameters and loads for EV2
}

object vehicle {
    parent electrical;
    name EV3;
    // Similar parameters and loads for EV3
}

object vehicle {
    parent electrical;
    name EV4;
    // Similar parameters and loads for EV4
}

object schedule {
    name Load_Profile_House1;
    // Define load profile schedule for House1
}

object schedule {
    name EV_Charging_Profile_EV1;
    // Define EV charging profile schedule for EV1
}

// Add similar schedule definitions for other houses and EVs
```

In the context of my research, a simplified GridLAB-D simulation to investigate V2G and G2V energy management strategies was developed. The simulation involves four houses and four EVs, aiming to optimize energy utilization and balance grid demand by leveraging EV battery energy.

Each house was treated as a module using the house object. Within each module, parameters specific to individual houses, such as the **heatgain_mode**, reflecting their energy consumption characteristics, were defined. Each house's energy load was simulated using **a ZIPload** object, allowing power consumption scheduling based on **Load_Profile_HouseX**, which was separately defined for each house.

Similarly, the modelling of EVs was executed using the vehicle object. Relevant parameters about EV behaviour were incorporated for each EV, and the scheduling of EV charging was based on **EV_Charging_Profile_EVX**, derived from the available data. The utilisation of schedules was implemented to replicate real-world load and charging behaviours. **Load_Profile_HouseX** represented the load consumption profile for each house, while **EV_Charging_Profile_EVX** represented the profile for EV charging. Incorporating these schedules facilitated the emulation of variations in power consumption and charging patterns throughout the simulated time.

The simulation time was also established using the clock module, where the start and stop times were specified while considering the UTC timezone, which can be adapted according to the user's requirements; this coding segment is presented as a simple example. This allowed for the control of the simulation's duration and the analysis of outcomes within the provided timeframe.

The provided GridLAB-D code presents a simplified simulation framework investigating V2G and G2V management strategies. By modelling four houses and four EVs, along with incorporating load and charging profiles, the objective was to demonstrate the potential utilization of EV battery energy to optimise grid power usage. This foundational code is a starting point for more comprehensive simulations and analyses, aligning with the research goal of exploring innovative energy management solutions using GridLAB-D. It is worth noting that this code has been simplified to illustrate the entire work's concept. Specific features were incorporated into this code to fulfil the research requirements for generating the dataset used in machine learning endeavours.

Appendix B

Model Evaluation and Performance Validations

This appendix presents a comprehensive overview of the evaluation and performance of machine-learning models. This section highlights the differences between the real and predicted values generated by various machine learning algorithms. The focus is on visualizing these differences through graphical representations that provide insights into the accuracy and reliability of the models.

Differences between Real and Predicted Values

This section showcases figures illustrating the discrepancies between the actual values and the predictions of different machine learning models. Visually depicting these differences valuable insights into how closely the models are aligned with the true values are gained visually depicting these differences gives valuable insights into how closely the models are aligned with the true values. It mentions that these 1200 samples are from 12000 testing data from all 50,000 in the first scenario. The reason for having only 1200 is to have a better visualization. This visualization shows what each ACC means before having any noises for each method.

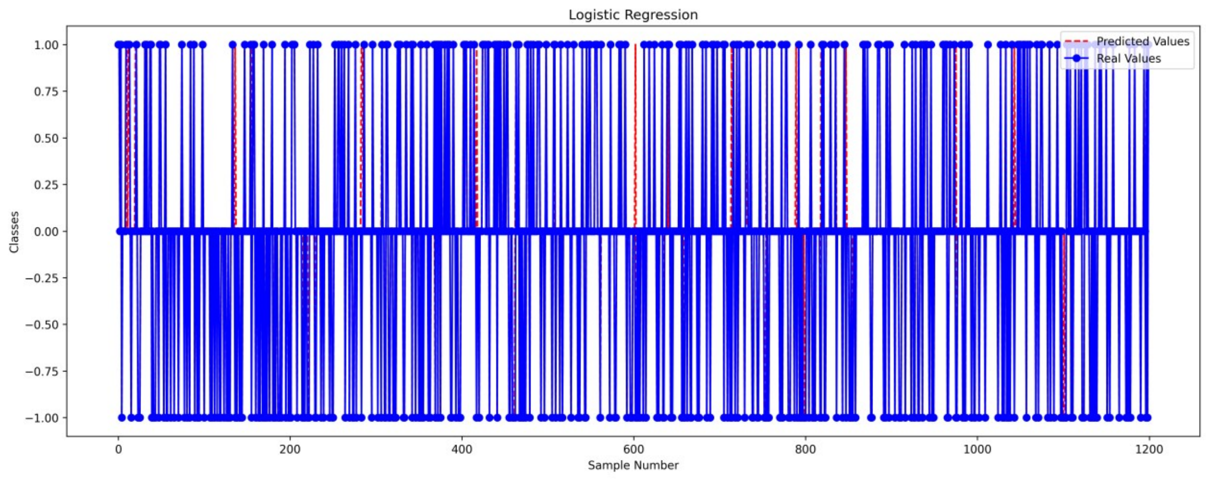


Figure 35 Real and predicted values of 1200 samples in Logistic Regression

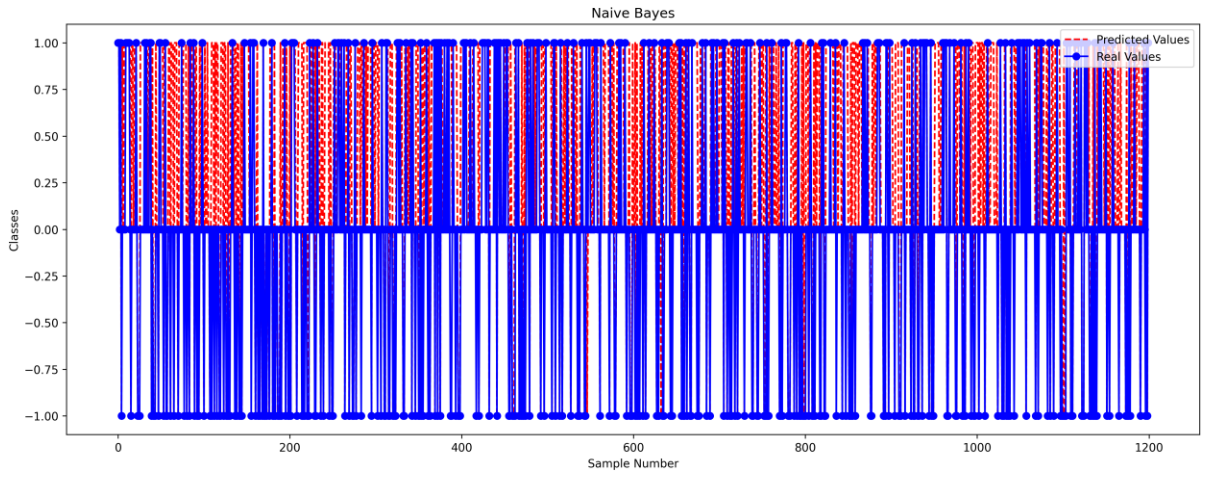


Figure 36 Real and predicted values of 1200 samples in Naïve Bayes

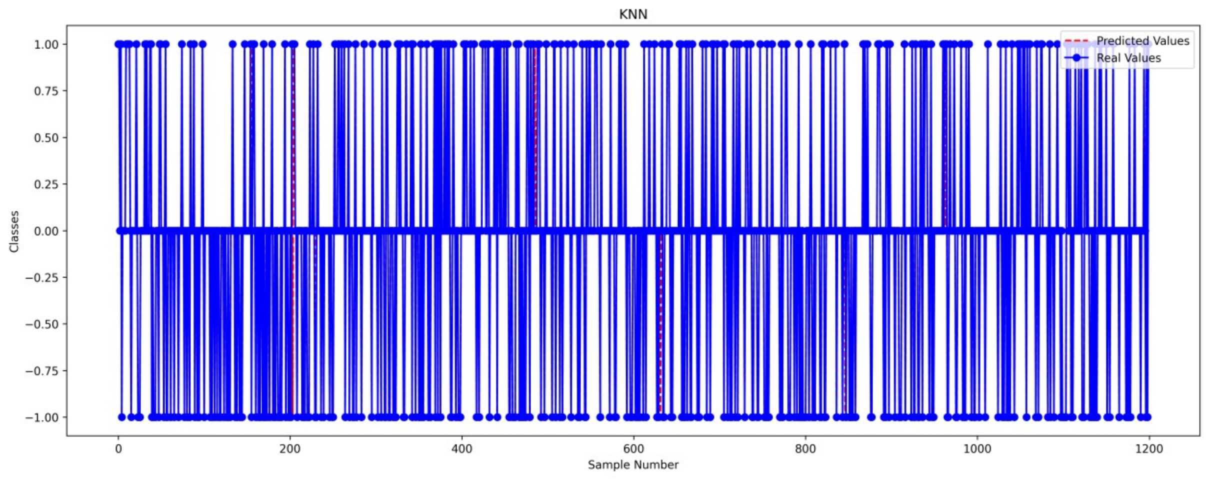


Figure 37 Real and predicted values of 1200 samples in KNN

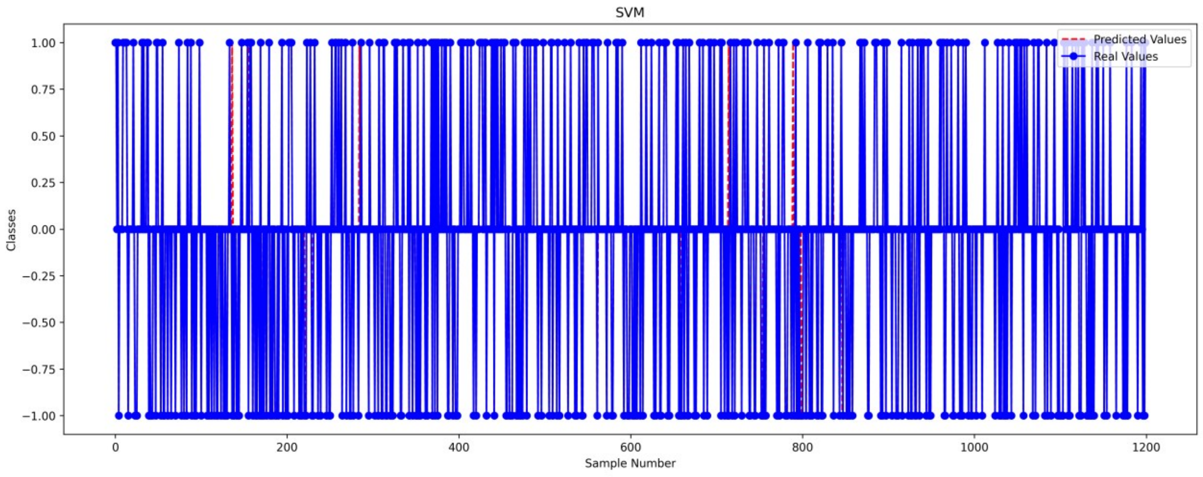


Figure 38 Real and predicted values of 1200 samples in SVM

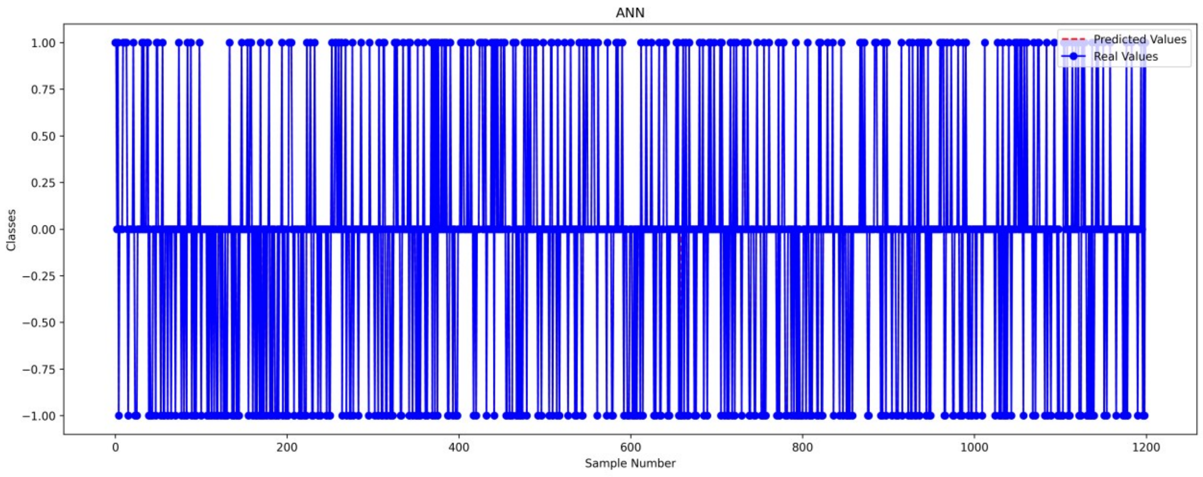


Figure 39 Real and predicted values of 1200 samples in SVM

Confusion Matrices

This part generated a collection of confusing matrices. These matrices represent the model's performance across different classes and corresponding predictions. The matrices are a diagnostic tool to assess how well the models classify and predict the different classes.

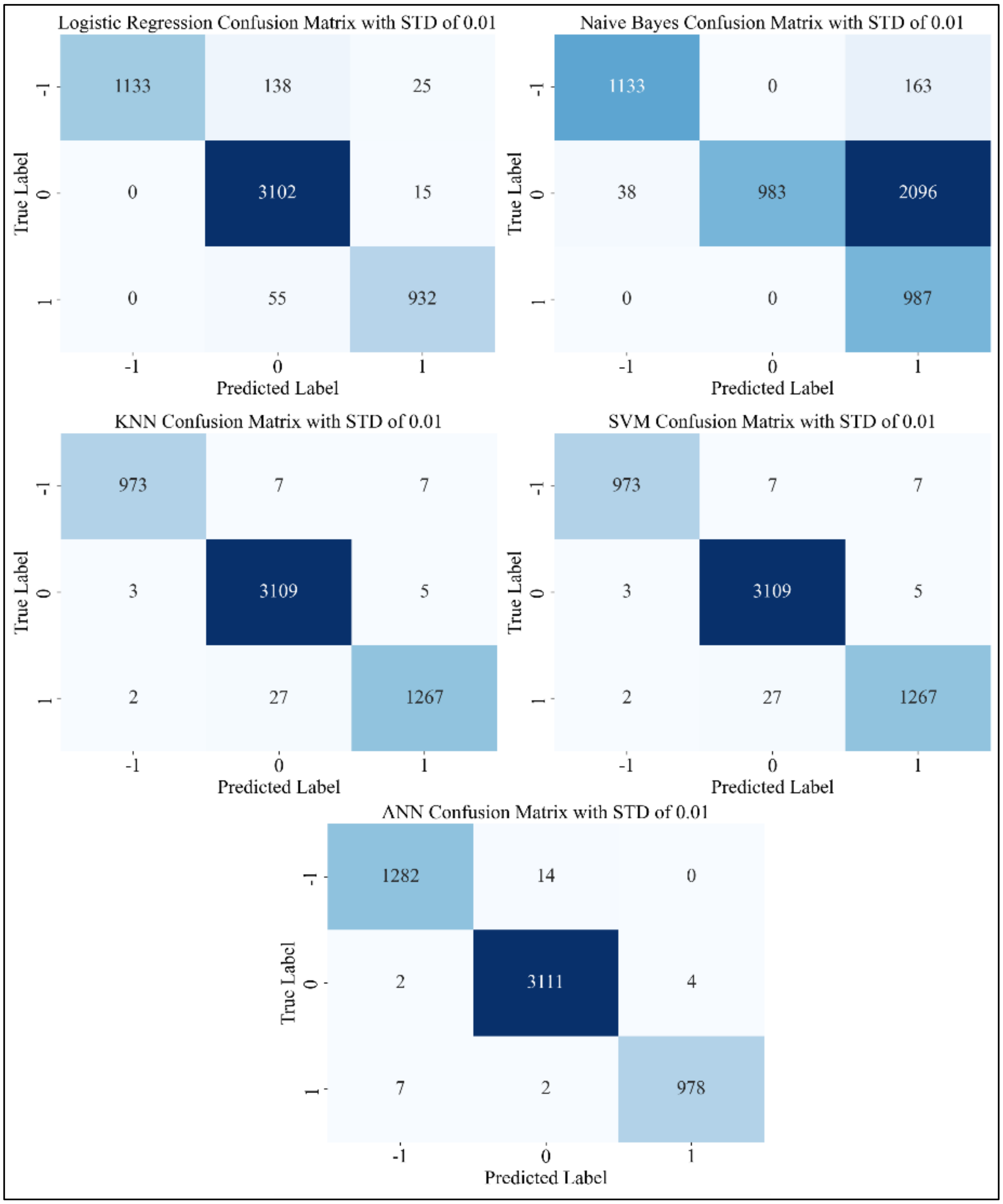
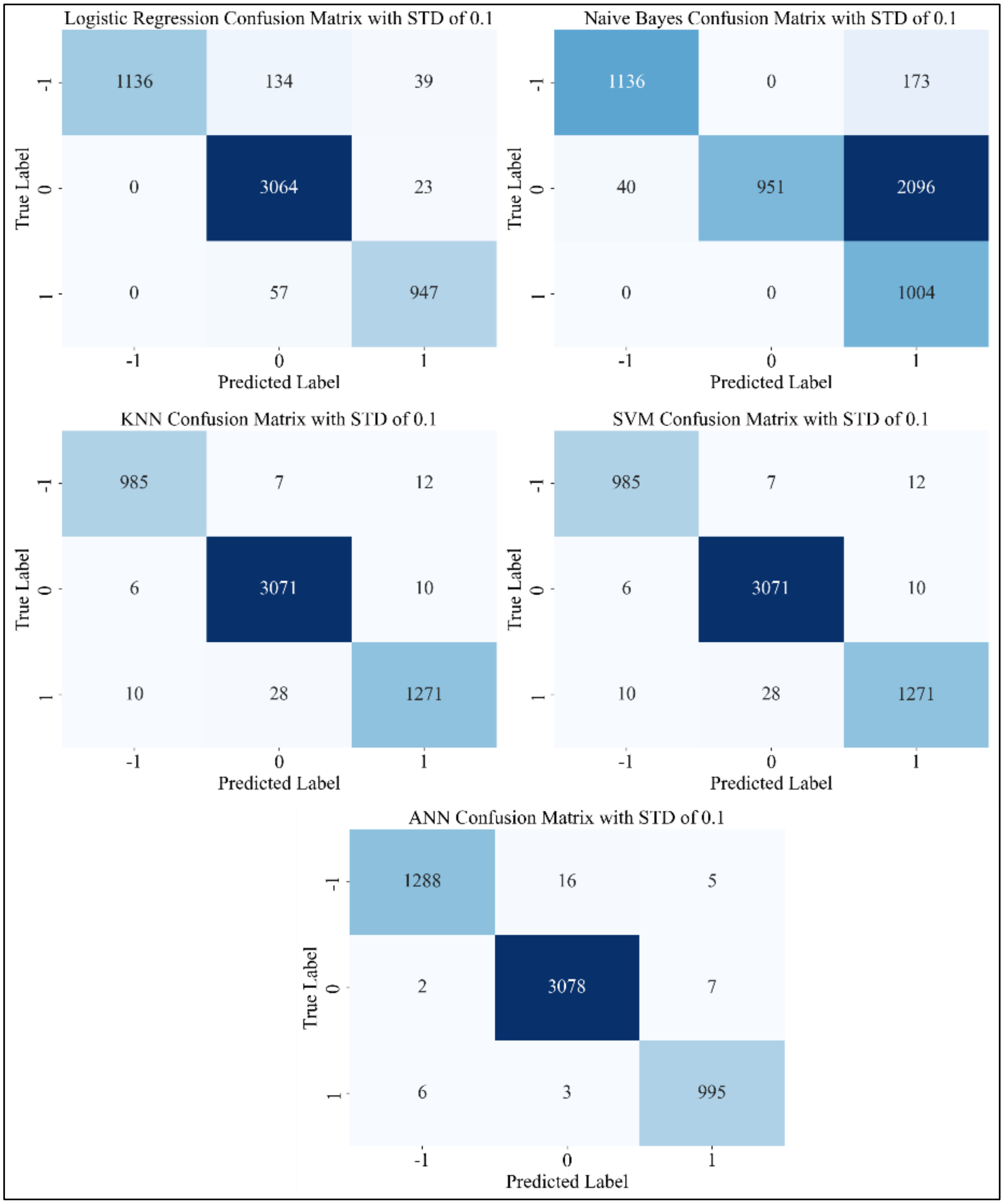


Figure 40 STD = 0.01



Logistic Regression	-1	0	1
-1	1136	134	39
0	0	3064	23
1	0	57	947

Naive Bayes	-1	0	1
-1	1136	0	173
0	40	951	2096
1	0	0	1004

KNN	-1	0	1
-1	985	7	12
0	6	3071	10
1	10	28	1271

SVM	-1	0	1
-1	985	7	12
0	6	3071	10
1	10	28	1271

ANN	-1	0	1
-1	1288	16	5
0	2	3078	7
1	6	3	995

Figure 41 STD = 0.1

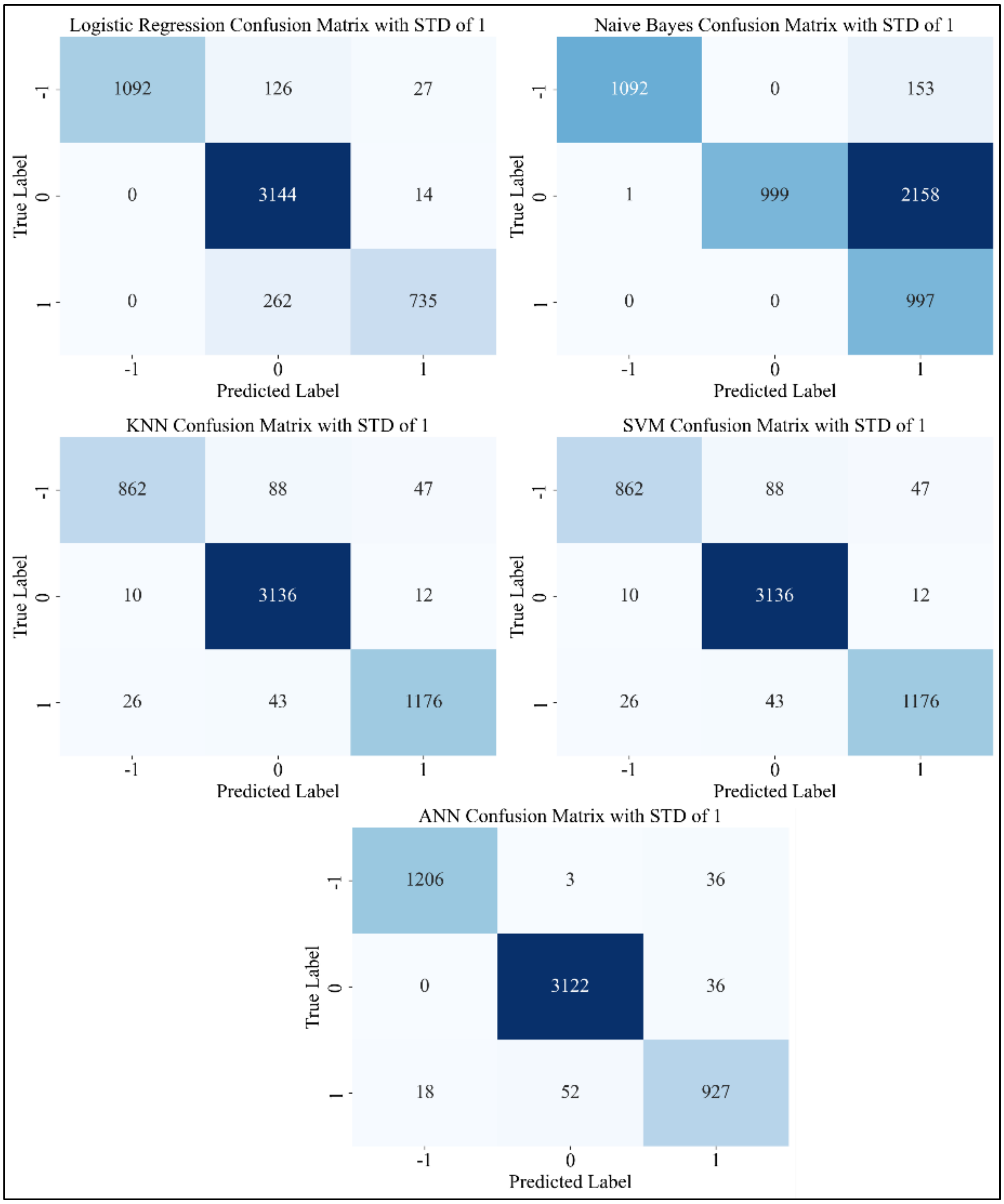


Figure 42 STD = 1

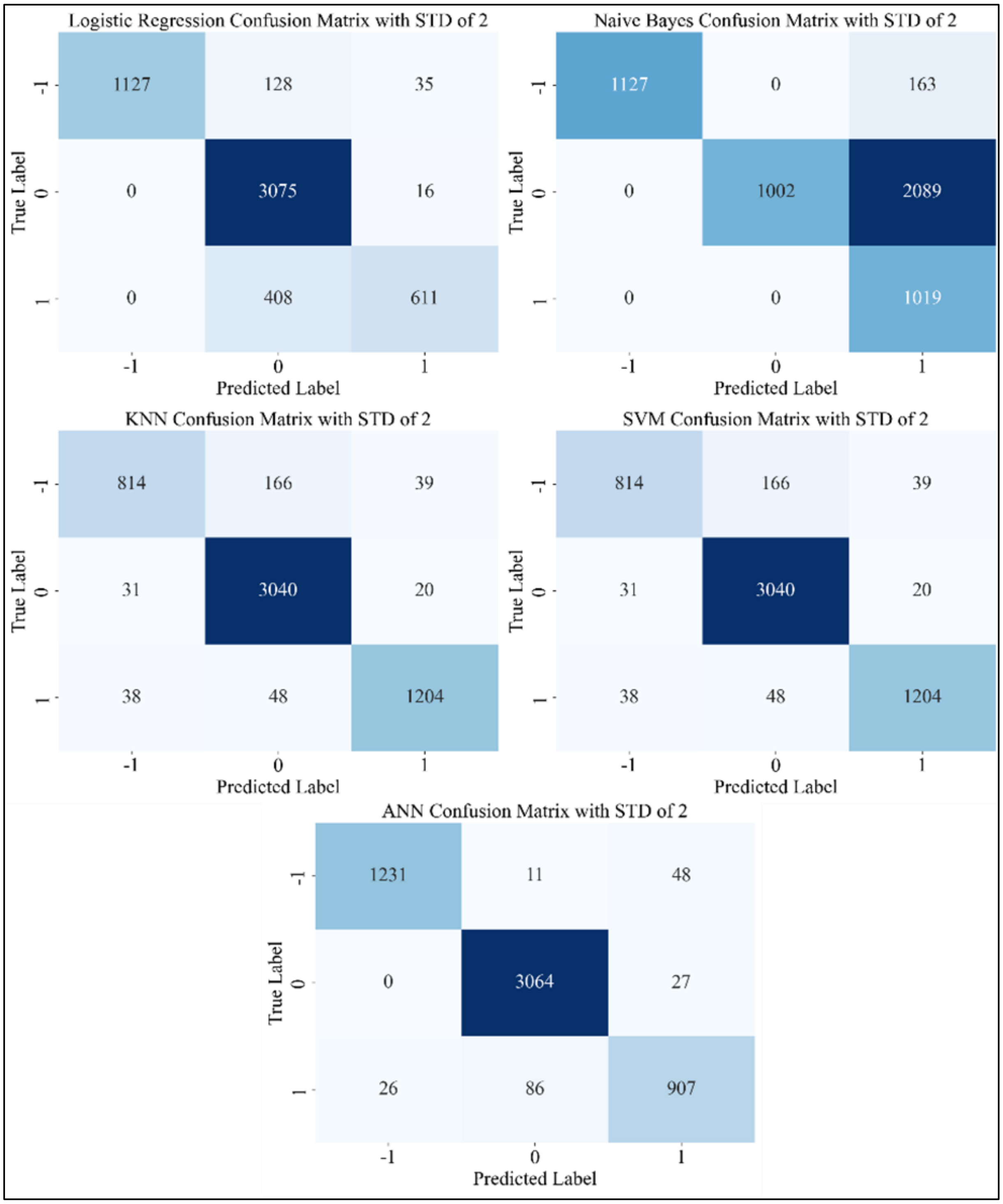


Figure 43 STD = 2

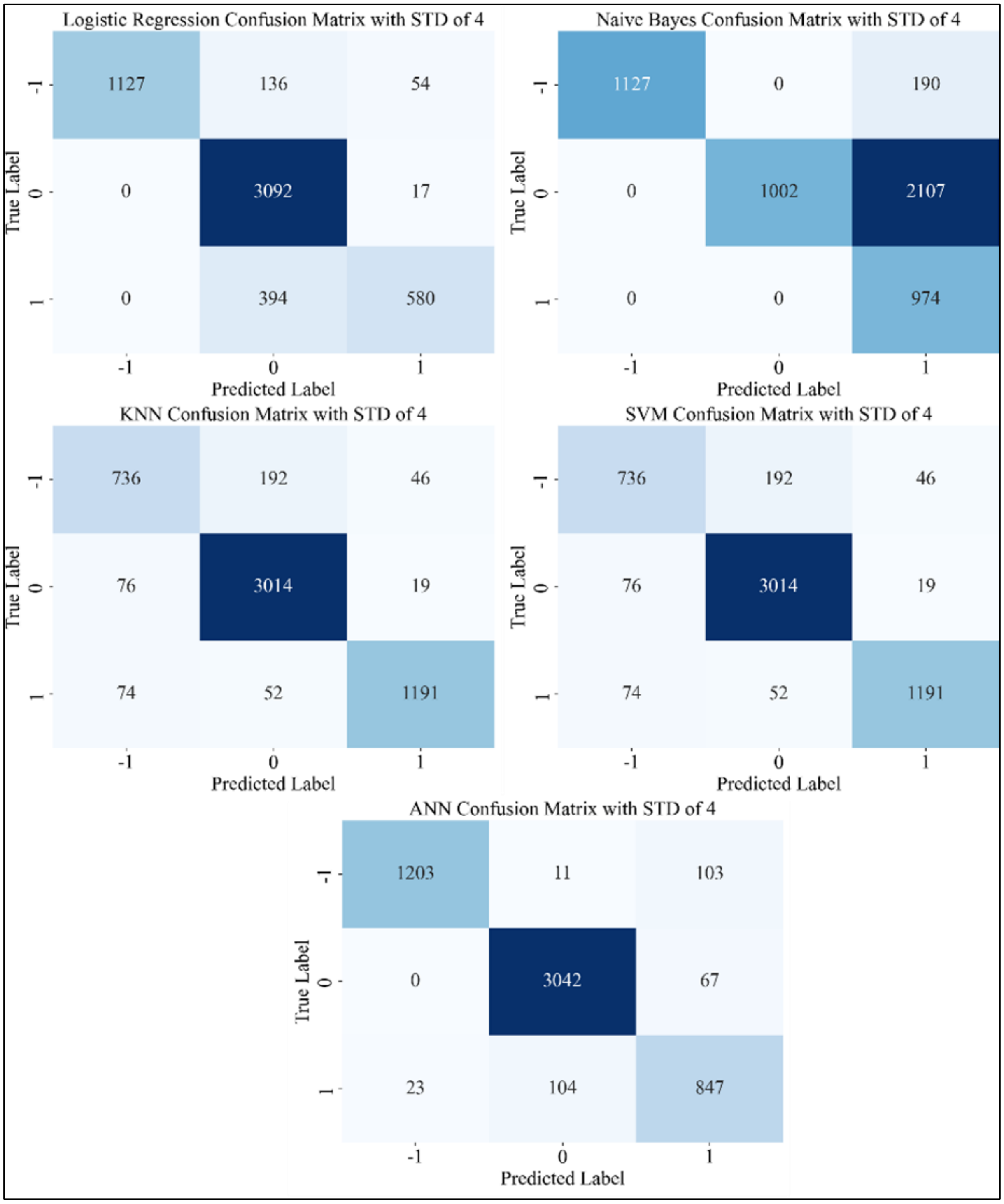


Figure 44 STD = 4

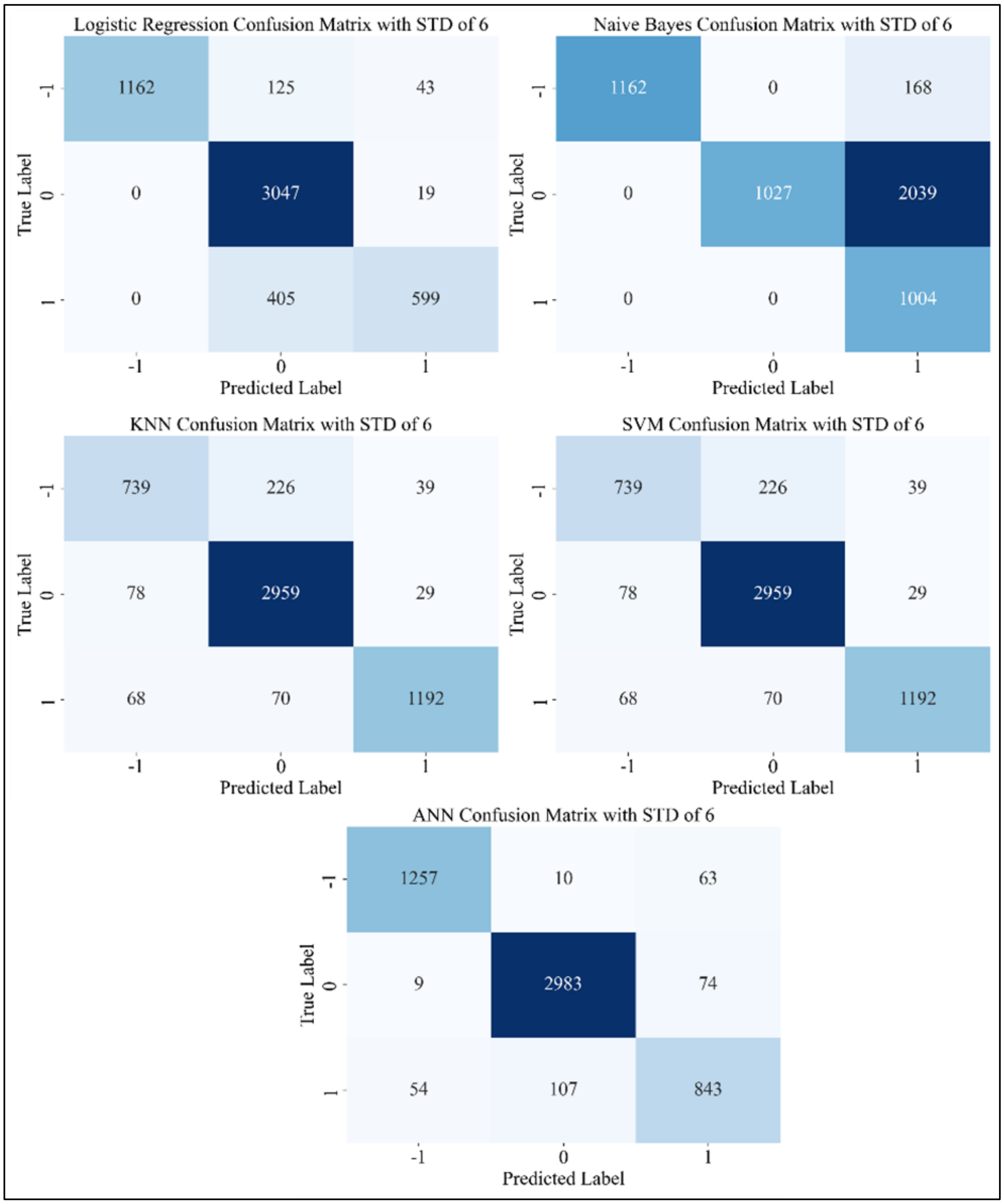


Figure 45 STD = 6