A Project Report on

Energy Consumption Prediction For Home Appliances Using Machine Learning

submitted in partial fulfillment for the award of

Bachelor of Technology

in

Computer Science & Engineering

by

S. L Supraja Devi (Y20ACS571) **S. Leela Krishna (Y20ACS552)**

SK.Mastan Vali(L21ACS418) **P.Naveen KumarReddy(Y20ACS526)**

Under the guidance of
Prof. K. Siva Kumar.

Department of Computer Science and Engineering
Bapatla Engineering College
(Autonomous)
(Affiliated to Acharya Nagarjuna University)
BAPATLA – 522 102, Andhra Pradesh, INDIA
2023-2024

Department of
Computer Science & Engineering

CERTIFICATE

This is to certify that the project report entitled **Energy Consumption Prediction For Home Appliances** that is being submitted by **S Lakshmi Supraja Devi(Y20ACS571)**, **S Leela Krishna (Y20ACS552), SK Matsan Vali (L21ACS418), P Naveen Kumar Reddy(Y20CS526)** in partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science & Engineering to the Acharya Nagarjuna University is a record of bonafide work carried out by them under our guidance and supervision.

Date:

Signature of the Guide
K Siva Kumar(ME)
Assist Prof.

Signature of the HOD
Dr. M Rajesh Babu
Prof. & HOD

DECLARATION

We declare that this project work is composed by ourselves, that the work contained herein is our own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

S Lakshmi Supraja Devi (Y20ACS571)
S Leela Krishna (Y20ACS552)
SK Mastan Vali (L21ACS418)
P Naveen Kumar Reddy (Y20ACS526)

Acknowledgement

We sincerely thank the following distinguished personalities who have given their advice and support for successful completion of the work.

We are deeply indebted to our most respected guide **K Siva Kumar, ME Assistant Professor,** Department of CSE, for his/her valuable and inspiring guidance, comments, suggestions and encouragement.

We extend our sincere thanks to **Dr. M Rajesh Babu, Prof. & Head** of the Dept. for extending his cooperation and providing the required resources.

We would like to thank our beloved Principal **Dr**. **Sk Nazeer** for providing the online resources and other facilities to carry out this work.

We would like to express our sincere thanks to our project coordinator **Dr. N. Sudhakar,** Prof. Dept. of CSE for his helpful suggestions in presenting this document.

We extend our sincere thanks to all other teaching faculty and non-teaching staff of the department, who helped directly or indirectly for their cooperation and encouragement.

S Lashmi Sipraja Devi (Y20ACS571)
S Leela Krishna (Y20ACS552)
SK Mastan Vali (L21ACS418)
P Naveen Kumar Reddy (Y20ACS526)

Table of Contents

List Of Figures

Abstract

The efficient management of energy consumption in residential settings is becoming increasingly vital for reducing electricity bills and mitigating environmental impact. This project aims to develop a data-driven solution for energy consumption prediction of home appliances using machine learning techniques. By collecting and analyzing real-time energy consumption data from smart meters and IoT devices, our system leverages advanced machine learning algorithms to forecast future energy usage patterns. The proposed system will empower users to make informed decisions about energy usage, optimize appliance schedules, and identify potential energy-saving opportunities. Through the integration of predictive models, data visualization tools, and user-friendly interfaces, our project seeks to provide a comprehensive solution for enhancing energy efficiency and sustainability in households.

1 Introduction

In an era where energy efficiency and sustainability are paramount concerns, the ability to accurately predict energy consumption becomes not just a convenience but a necessity. The surge in smart home technology has revolutionized the way we interact with our living spaces, offering unprecedented levels of control and insight into our energy usage. Leveraging the power of machine learning, our project aims to further enhance this capability by developing a predictive model for energy consumption of home appliances.

1.1 Machine Learning

Machine learning is a subset of artificial intelligence (AI) that focuses on the development of algorithms and statistical models that enable computers to learn from and make predictions or decisions based on data. The fundamental idea behind machine learning is to enable computers to learn and improve from experience without being explicitly programmed for every task.

In traditional programming, developers write explicit instructions telling the computer how to solve a particular problem. In contrast, in machine learning, instead of programming specific rules, algorithms are trained on large datasets to recognize patterns and relationships within the data. These algorithms iteratively learn from the data, refining their predictions or decisions over time.

1.1.1 Supervised Machine Learning

Supervised machine learning is a type of machine learning where the algorithm is trained on a labelled dataset. In a supervised learning setting, each example in the training dataset consists of input features and corresponding output labels. The goal of supervised learning is to learn a mapping from the input features to the output labels so that the algorithm can make accurate predictions on unseen data.

1.1.2 Unsupervised Machine Learning

Unsupervised machine learning is a type of machine learning where the algorithm is trained on data without labelled responses. Unlike supervised learning, where the algorithm learns from labelled examples, unsupervised learning algorithms must discover patterns and structures within the data on their own.

The primary goal of unsupervised learning is to explore and uncover hidden patterns, relationships, or structures within the dataset. This can include identifying clusters of similar data points, reducing the dimensionality of the data, or detecting anomalies or outliers

1.2 Introduction to Consumption of Electricity

Electricity is vital for powering homes, businesses, industries, and essential infrastructure like hospitals and schools. It enables modern conveniences, drives economic activity, and improves living standards.

Over time, electricity consumption has consistently risen due to factors such as population growth, urbanization, and technological progress. Patterns include seasonal variations, daily peaks (often corresponding to peak hours of activity), and long-term trends influenced by economic cycles and technological advancements.

1.2.1 Electricity Consumption

Electricity consumption refers to the amount of electrical energy used over a specific period of time. It is measured in units such as kilowatt-hours (kWh) and is typically recorded by electricity meters installed at homes, businesses, industries, and other facilities.

Electricity consumption is a fundamental aspect of modern life, powering various activities and devices, including lighting, heating, cooling, electronics, appliances, machinery, and transportation (in the case of electric vehicles). It plays a crucial role in economic development, social welfare, and quality of life.

1.2.1.1 Generating Electricity

Generating electricity involves the process of converting various forms of energy into electrical energy. Electricity generation is a critical component of the energy sector, providing the power necessary to fuel modern societies, industries, and technologies. Here's an overview of the process:

Population and Urbanization: Higher population densities and urbanization tend to result in increased electricity consumption due to greater demand for residential, commercial, and industrial energy services.

Economic Activity: Economic growth and industrial development drive electricity demand, as industries, businesses, and households require more energy to operate machinery, equipment, and appliances.

Climate and Weather: Climate conditions influence electricity usage for heating, cooling, and lighting, with higher demand during extreme temperatures.

1.2.1.2 Electricity Storage

Storing electricity, also known as energy storage, refers to the process of capturing and storing electrical energy for later use. Energy storage plays a crucial role in modern energy systems by enabling the balancing of supply and demand, managing fluctuations in renewable energy generation, improving grid stability, and enhancing overall energy efficiency.

1.2.1.3 Consumption of Electricity

Consumption of electricity refers to the amount of electrical energy used by consumers over a specific period of time. It is a measure of the total electricity consumed by households, businesses, industries, and other entities within a given region or system.

Consumption of electricity is typically measured in units such as kilowatt-hours (kWh), which represent the amount of energy consumed by using 1 kilowatt (kW) of power for 1 hour. For example, if a device with a power rating of 1 kW is used continuously for 1 hour, it would consume 1 kWh of electricity.

1.3 Objective

Accuracy Improvement: Continuously refine and optimize the predictive models to improve their accuracy and reliability, ensuring that they can effectively forecast energy consumption under various operating conditions.

2 Literature Survey

Understanding Energy Consumption Patterns:

- Identifying the energy consumption patterns of different home appliances.
- Analyzing the factors that influence the energy consumption of home appliances.

Predicting Energy Consumption:

- Developing a regression model to predict the energy consumption of home appliances.
- Evaluating the accuracy and reliability of the energy consumption prediction model.

The literature surrounding energy consumption prediction using machine learning techniques is extensive and multifaceted. Researchers have delved into various methodologies, leveraging historical energy usage data alongside weather conditions and other contextual factors to develop accurate predictive models. One prominent area of study is Non-Intrusive Load Monitoring (NILM), which focuses on disaggregating household energy consumption data to individual appliance-level information. In this realm, algorithms and techniques have been developed to identify and track specific appliance energy usage within households, enabling more detailed predictions. Moreover, machine learning algorithms like random forest, support vector machines, neural networks, and ensemble methods have been extensively explored for energy consumption prediction tasks. Comparative studies have evaluated these algorithms, shedding light on their strengths and limitations. Feature engineering, another critical aspect, involves selecting relevant input variables to capture underlying energy usage patterns. Various techniques, including time-series decomposition and statistical aggregation, have been employed for this purpose. Despite progress, challenges like data quality issues and model interpretability persist. Real-world applications in smart homes and energy-efficient buildings demonstrate the practical utility of predictive models in optimizing energy usage and promoting sustainability. Looking forward, interdisciplinary collaborations and emerging technologies like IoT and big data analytics offer exciting avenues for further research and innovation in energy consumption prediction. Our project aims to contribute to this ongoing advancement by synthesizing existing literature and proposing novel methodologies and approaches.

3 Problem Statement

Rising energy costs and concerns about environmental sustainability necessitate a focus on improving energy efficiency in homes.

Home appliances are major contributors to household energy consumption, but accurately predicting their individual usage remains a challenge.

Traditional methods for estimating appliance energy use often lack precision.

Disaggregating overall household energy consumption data to isolate individual appliance usage can be complex.

This project aims to develop a machine learning model capable of accurately predicting the energy consumption of individual home appliances.

Analyze and pre-process historical energy consumption data, including appliance-specific data if available, alongside relevant environmental and temporal features.

Develop and implement machine learning algorithms to predict appliance energy consumption.

A robust and accurate machine learning model for predicting appliance energy consumption.

Improved understanding of the factors influencing appliance energy usage.

Potential for developing user-friendly applications that empower consumers to:

Track and visualize individual appliance energy consumption.

Identify opportunities to reduce energy use and costs.

Make informed decisions about appliance usage patterns.

This project contributes to the development of intelligent and sustainable homes by enabling better management of appliance energy consumption. The findings can benefit not only individual consumers but also utilities through improved demand response strategies and integration of renewable energy sources.

4 System Analysis

Define the problem and objectives: Clearly define the problem and the objectives of the energy consumption prediction system. This includes identifying the stakeholders, the scope of the system, and the desired outcomes.

Gather and analyze data: Collect historical energy consumption data and related weather information. This data should include temperature, humidity, illumination, and time of day. Analyze the data to identify patterns and trends in energy consumption.

Preprocess the data: Clean and preprocess the data to ensure that it is in a format that can be used for machine learning. This includes handling missing values, scaling the data, and transforming the data into a format that can be used for time series forecasting.

Develop the machine learning model: Develop a machine learning model using a time series forecasting approach. This model should be able to accurately forecast energy usage based on the supplied features.

Train and validate the model: Train the machine learning model using historical energy consumption data and related weather information. Validate the model using a separate dataset to ensure that it is able to accurately forecast energy usage.

Evaluate the model: Evaluate the performance of the machine learning model using metrics such as mean absolute error, mean squared error, and R-squared.

Deploy the model: Deploy the machine learning model in a production environment where it can be used to forecast energy usage.

Monitor and maintain the model: Monitor the performance of the machine learning model and make adjustments as necessary. This includes retraining the model with new data and updating the model to reflect changes in energy consumption patterns.

Continuously improve the system: Continuously improve the energy consumption prediction system by incorporating feedback from stakeholders and incorporating new data and technologies.

4.1 Existing System

The results of the Ecoisme system, including the accuracy of its energy consumption predictions and the energy-saving recommendations provided to users. Highlight that applying the multitarget classification algorithm of Random k-labELsets by disjoint subsets with the decision tree resulted in F-score and accuracy values greater than 89% for high-power loads such as A/C and water heater. Additionally, mention that the normalized error values of power prediction outperformed the use of Factorial Hidden Markov Model and binary state modeling system.

4.1.1 Limitations

The accuracy of the predictions depends on the quality of the collected data. The accuracy of the Ecoisme system's predictions can be influenced by factors such as noise in the collected data, measurement errors, and incorrect appliance categorization.

The current approach of using machine learning algorithms may not be suitable for all types of home appliances. For example, it may not be feasible to accurately predict the energy consumption of appliances that operate on complex algorithms or have irregular energy consumption patterns.

The current implementation of the Ecoisme system uses only machine learning algorithms for energy consumption prediction. Incorporating other types of data analysis techniques, such as Factorial Hidden Markov Model and binary state modeling system, could potentially improve the system's accuracy.

The Ecoisme system may not be compatible with all types of smart meters. Users may need to invest in compatible hardware or upgrade their existing smart meters to ensure compatibility with the system.

4.2 Proposed System

This system leverages a machine learning approach, specifically a Random Forest algorithm, to predict the energy consumption of individual home appliances. It aims to achieve a high level of accuracy (targeting 95%) in its predictions.

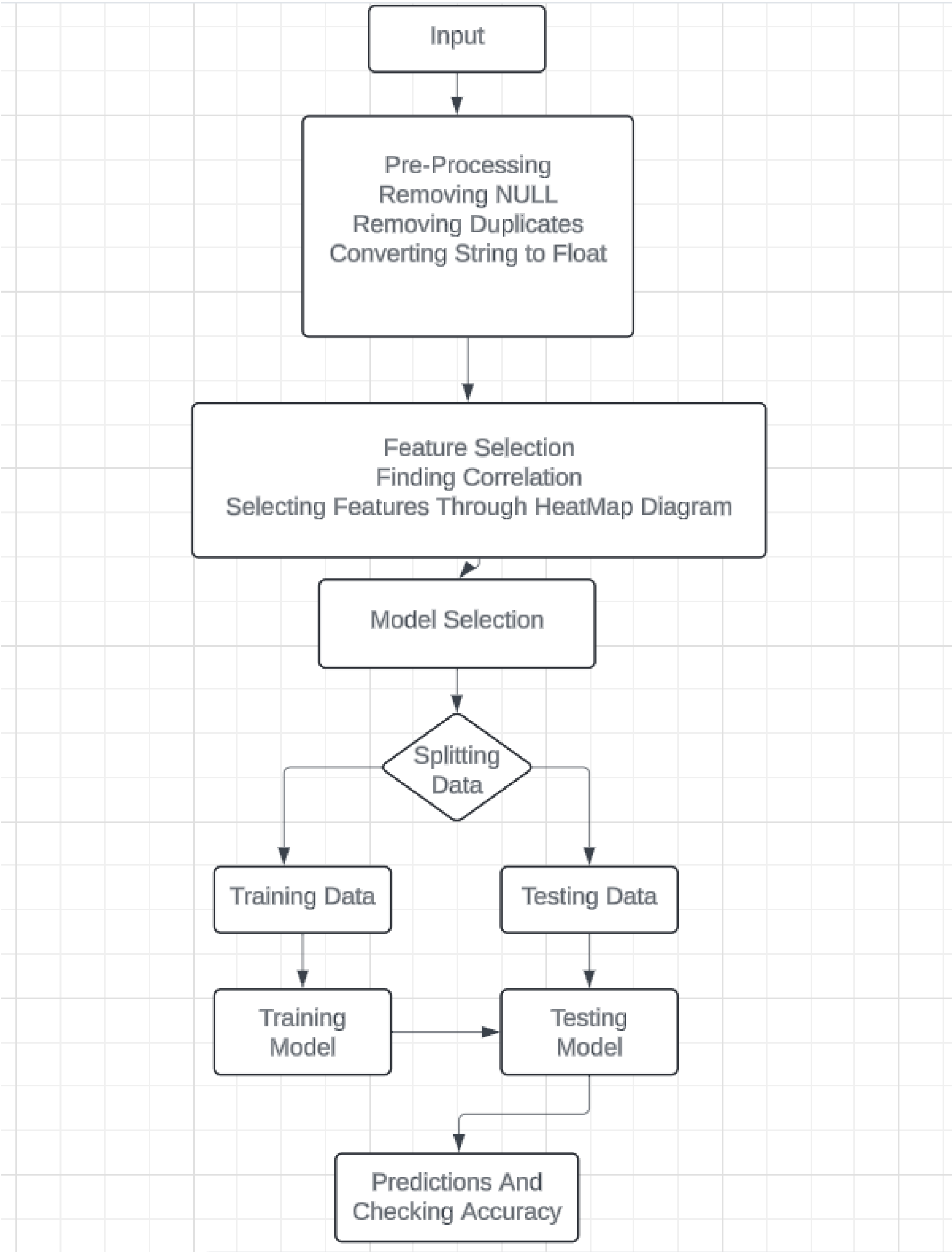
Random Forest algorithm due to its robustness to overfitting, ability to handle mixed-data types, and interpretability of results.

The accuracy of the model heavily relies on the quality and quantity of available data. Ensure your data is clean, representative, and covers diverse usage patterns.

This proposed system leverages a Random Forest model to predict appliance energy consumption with a target accuracy of 95%.

4.3 Architecture

The architecture of the energy consumption prediction system for home appliances using machine learning consists of several interconnected components designed to collect, process, analyze, and visualize energy consumption data. The architecture follows a modular and scalable design to accommodate varying data volumes, computational requirements, and user interactions. Below are the key components and their functionalities:



4.3.1 Architecture of Proposed System

4.3.1 Modules Used in Proposed System

The System Implementation uses RandomForest and XGBoost to detect and finding the Predict the Approximate Appliances consumed by household energy. The Proposed System contains the following modules.

Pre-processing Module: The pre-processing module typically includes various data cleaning, transformation, and normalization techniques to ensure that the input data is of high quality, consistent, and ready for analysis. This may involve tasks such as handling missing values, removing outliers, scaling numerical features, encoding categorical variables, and creating new derived features.

Feature Selection: One popular technique for feature selection is the use of a heatmap, which visually represents the correlation between different features in the dataset. A heatmap provides a color-coded matrix where each cell represents the correlation coefficient between two features. A high correlation coefficient indicates a strong linear relationship between two features, while a low correlation coefficient suggests little to no relationship.

5 Methodology

This section outlines the methodological approach for developing a system that utilizes a Random Forest algorithm to predict home appliance energy consumption with a target accuracy of 95%.

5.1 Dataset

The date sets can be downloaded from Kaggle .

There are 29 features to describe appliances energy use :

1. date : time year-month-day hour:minute:second

2. lights : energy use of light fixtures in the house in Wh

3. T1 : Temperature in kitchen area, in Celsius

4. T2 : Temperature in living room area, in Celsius

5. T3 : Temperature in laundry room area

6. T4 : Temperature in office room, in Celsius

7. T5 : Temperature in bathroom, in Celsius

8. T6 : Temperature outside the building (north side), in Celsius

9. T7 : Temperature in ironing room, in Celsius

10.T8 : Temperature in teenager room 2, in Celsius

11. T9 : Temperature in parents' room, in Celsius

12. T_out : Temperature outside (from Chievres weather station), in Celsius

13. Tdewpoint : (from Chievres weather station), Â°C

14. RH_1 : Humidity in kitchen area, in %

15. RH_2 : Humidity in living room area, in %

16. RH_3 : Humidity in laundry room area, in %

17. RH_4 : Humidity in office room, in %

18. RH_5 : Humidity in bathroom, in %

19. RH_6 : Humidity outside the building (north side), in %

20. RH_7 : Humidity in ironing room, in %

21. RH_8 : Humidity in teenager room 2, in %

22. RH_9 : Humidity in parents' room, in %

23. RH_out :Humidity outside (from Chievres weather station), in %

24. Pressure : (from Chievres weather station), in mm Hg

25. Wind speed: (from Chievres weather station), in m/s

26. Visibility :(from Chievres weather station), in km

27. Rv1 :Random variable 1, non-dimensional

28. Rv2 :Random variable 2, non-dimensional

29. Appliances : Total energy used by appliances, in Wh.

5.2 Pre-Processing

In machine learning, data pre-processing is a crucial step that involves manipulating, cleaning, and transforming raw data to prepare it for analysis by machine learning models. It's like tidying up your workspace before you start building something – the cleaner and more organized your data is, the easier it will be for your model to learn from it and make accurate predictions.

5.2.1 Data Cleaning

Utilize techniques to handle missing values (e.g., imputation methods), identify and address outliers, and ensure data consistency across different sources.

5.2.2 Feature Selection

Choose the most relevant features to improve model performance and avoid overfitting. Techniques like correlation analysis or feature importance scores from the Random Forest model itself can be used for selection. This step helps identify the most informative features for predicting appliance energy consumption.

5.2.3 Feature Engineering

Create new features based on existing data that might influence appliance energy consumption. Examples include:

Time-based features (e.g., hour of the day, day of the week, season)

Appliance state (on/off)

Environmental factors (temperature, humidity) (if relevant)

5.2.4 Normalization

Normalization is a technique used in data pre-processing for machine learning tasks. It involves transforming features in your dataset to a common scale. This is important because machine learning algorithms can be sensitive to the scale of the data. Features with larger scales can have an outsized influence on the model's learning process compared to features with smaller scales. Normalization helps create a level playing field for all features, ensuring they contribute equally during model training.

5.2.5 Data Splitting

Data splitting helps address this issue by creating separate sets for training and testing:

Training Set: The largest portion of the data (usually 60% to 80%) is used to train the machine learning model. The model learns from the patterns and relationships present in this data.

Testing Set: This set (usually 20% to 40% of the data) is held out and not used during training. Once the model is trained, it's evaluated on the testing set to assess its generalizability and performance on unseen data. This provides a more realistic measure of how well the model will perform in real-world scenarios.

5.3 Algorithms

An algorithm, in a nutshell, is a set of step-by-step instructions that tells a computer how to perform a specific task. It's like a recipe for a computer program, breaking down a complex problem into manageable, logical steps. Algorithms are fundamental to all computer science and play a vital role in various fields.

5.3.1 Random Forest

Random Forest is a powerful machine learning algorithm that excels at both classification and regression tasks. It's a popular choice for various applications, including our project of predicting appliance energy consumption.

Random Forest doesn't rely on a single decision tree but rather on a multitude of them, hence the "forest" analogy. This ensemble approach helps reduce overfitting and improve the model's overall accuracy and generalizability.

Building the Forest:

Random Subsets of Data: During training, the algorithm creates multiple subsets of your data (with replacement, meaning a data point can appear in multiple subsets). This process introduces diversity among the trees.

Growing Decision Trees: Each data subset is used to build a separate decision tree. These trees can have different depths and random subsets of features considered at each split. This further increases diversity within the forest.

Making Predictions: When a new, unseen data point arrives for prediction:

Each tree in the forest makes a prediction based on its own learned rules.

The final prediction is the majority vote (for classification) or the average (for regression) of the individual tree predictions.

High Accuracy: By combining predictions from multiple trees, Random Forest can achieve high accuracy on various tasks.

Robustness to Overfitting: The random nature of tree building and subset selection helps prevent the model from memorizing specific patterns in the training data, leading to better performance on unseen data.

Handling Missing Data: Random Forest can inherently handle missing values to some extent, as different trees might use different features for splitting.

Interpretability: Unlike some complex models, Random Forest offers some level of interpretability through feature importance scores. These scores indicate which features have the most significant influence on the model's predictions, providing insights into the factors affecting appliance energy consumption in your case.

5.3.2 XGBoost

XGBoost (eXtreme Gradient Boosting) is a powerful machine learning algorithm specifically designed for regression and classification tasks. It's known for its efficiency, scalability, and ability to achieve state-of-the-art performance on many problems, including potentially your appliance energy prediction project. Here's a deeper dive into XGBoost:

Building on Gradient Boosting:

XGBoost belongs to the family of gradient boosting algorithms. These algorithms work in a stage-wise fashion, building an ensemble of weak learners (often decision trees) sequentially. Each new learner focuses on improving the predictions of the previous ones by fitting to the residuals (errors) of the prior model.

5.4 Evaluation of Algorithms

Evaluating machine learning algorithms is a crucial step in the development process. It helps you assess the performance of different models and choose the one that best suits your needs for appliance energy prediction. Here's a breakdown of key metrics and techniques for algorithm evaluation:

Performance Metrics: The choice of metrics depends on whether you're dealing with a classification or regression task. For appliance energy prediction, which is a regression problem, here are some common metrics:

Mean Squared Error (MSE): This metric measures the average squared difference between the predicted values and the actual energy consumption values. Lower MSE indicates better model performance, signifying the model's ability to make predictions close to the real values.

Root Mean Squared Error (RMSE): The square root of MSE. It's easier to interpret in the same units as your target variable (energy consumption in kWh). Lower RMSE indicates better performance.

Mean Absolute Error (MAE): This metric calculates the average absolute difference between predicted and actual values. It's less sensitive to outliers compared to MSE. Lower MAE suggests better performance.

R-squared: This metric represents the proportion of variance in the target variable (energy consumption) that your model can explain. A value closer to 1 indicates a better fit, meaning the model explains most of the variations in energy consumption.

6 System Requirements and Specifications

The success of any software project hinges on the clarity and precision of its requirements and specifications. System requirements and specifications serve as the foundation upon which the entire development process is built, guiding stakeholders, developers, and testers throughout the project lifecycle. By defining the functional and non-functional aspects of the system, requirements and specifications ensure alignment between user expectations and the final product. On a broader scale, the System Requirements Document (SRD) incorporates both hardware and software elements that make up the entire system. This document provides a functional overview describing the system's overall purpose and the functionalities it delivers.

6.1 Hardware Requirements

For a project focused on energy consumption prediction for home appliances using machine learning, the hardware components required may vary depending on the specific implementation and deployment scenario. However, here are some typical hardware components that may be involved.

Smart Meter: Smart meters are essential hardware devices that measure and record electricity consumption at regular intervals. These meters typically have communication capabilities (e.g., Wi-Fi, Zigbee, or cellular) to transmit consumption data to a central server or cloud platform.

Data Processing Hardware: High-performance CPUs, GPUs, or specialized accelerators (e.g., TPUs) may be used to train machine learning models and perform inference tasks.

CPU: A multi-core processor (minimum 4 cores) with decent clock speed (e.g., 3.0 GHz or higher) is recommended. Consider higher core counts and clock speeds for complex models or real-time predictions.

RAM: Allocate at least 8GB of RAM for smooth operation. More RAM (16GB or higher) might be beneficial for handling large datasets or complex models.

Storage: Solid State Drive (SSD) with sufficient storage space is preferred for faster data access and processing, especially when dealing with large datasets.

Graphics Processing Unit (GPU): While not always mandatory, a GPU can significantly accelerate training times for complex deep learning models.

6.2 Software Requirements

The software requirements document serves as a critical blueprint for the development, testing, and deployment of a software project. It outlines the functional and non-functional specifications that define the behavior, features, and performance characteristics of the software system. By clearly articulating the needs and expectations of stakeholders, the software requirements document provides a roadmap for the development team to follow, ensuring alignment between user requirements and the final product.

These requirements are the result of comprehensive analysis and collaboration among stakeholders, including users, business analysts, and development teams. They encapsulate the essential features, functionalities, and constraints that shape the design and implementation of the software system.

6.2.1 Software Libraries

Pandas: It offers functionalities for data cleaning, transformation, and analysis, making it essential for pre-processing raw energy consumption data.

NumPy: NumPy is a fundamental library for numerical computing in Python, providing support for multidimensional arrays and mathematical operations.It is often used in conjunction with Pandas for efficient data manipulation and computation, especially when working with large datasets.

Scikit-learn: Scikit-learn is a versatile machine learning library that offers a wide range of algorithms and tools for building predictive models. It provides implementations for various regression and classification algorithms, including decision trees, random forests, support vector machines (SVM), and gradient boosting.

Matplotlib and Seaborn: Matplotlib and Seaborn are visualization libraries used for creating static, interactive, and publication-quality plots and charts.They allow you to visualize energy consumption trends, model predictions, and evaluation metrics to gain insights from data analysis results.

Flask: Flask are web application frameworks for building web-based user interfaces and APIs. You can use these frameworks to develop user interfaces for accessing energy consumption predictions, configuring system settings, and visualizing data insights.

Statsmodels: Statsmodels is a library for statistical modeling and hypothesis testing, providing tools for regression analysis, time series analysis, and more.

OS: The os module allows you to manipulate files and directories on the file system. You can create, delete, rename, and modify files and directories using functions like os.path.join(), os.listdir(), os.mkdir(), os.remove(), os.rename(), etc.

TensorFlow: TensorFlow supports various deployment options for deploying trained models to production environments. This includes TensorFlow Serving for serving models via RESTful APIs, TensorFlow Lite for deploying models on mobile and edge devices, and TensorFlow.js for running models in web browsers.

Zipfile: You can use the ZipFile class to read and extract files from existing ZIP archives. The ZipFile object provides methods like namelist() to list the contents of the archive, read() to read the contents of a specific file, and extract() to extract files or directories from the archive to a specified location on the filesystem.

Dateutil: One of the primary features of the dateutil module is its ability to parse date and time strings in various formats, including ISO 8601, RFC 2822, and many others. The parse() function automatically detects the format of the input string and converts it into a datetime object.

Time: The time module provides functions to access the current time in various formats. The time() function returns the current system time in seconds since the epoch (January 1, 1970, 00:00:00 UTC), while gmtime() and localtime() functions return the current time in structured form as a time.struct_time object in UTC and local time zone, respectively.

7 System Design

System design for our project, "Energy Consumption Prediction for Home Appliances using Machine Learning," involves architecting a robust and scalable framework capable of efficiently processing data, training machine learning models, and making predictions. Here's an overview of the key components and considerations for the system design.

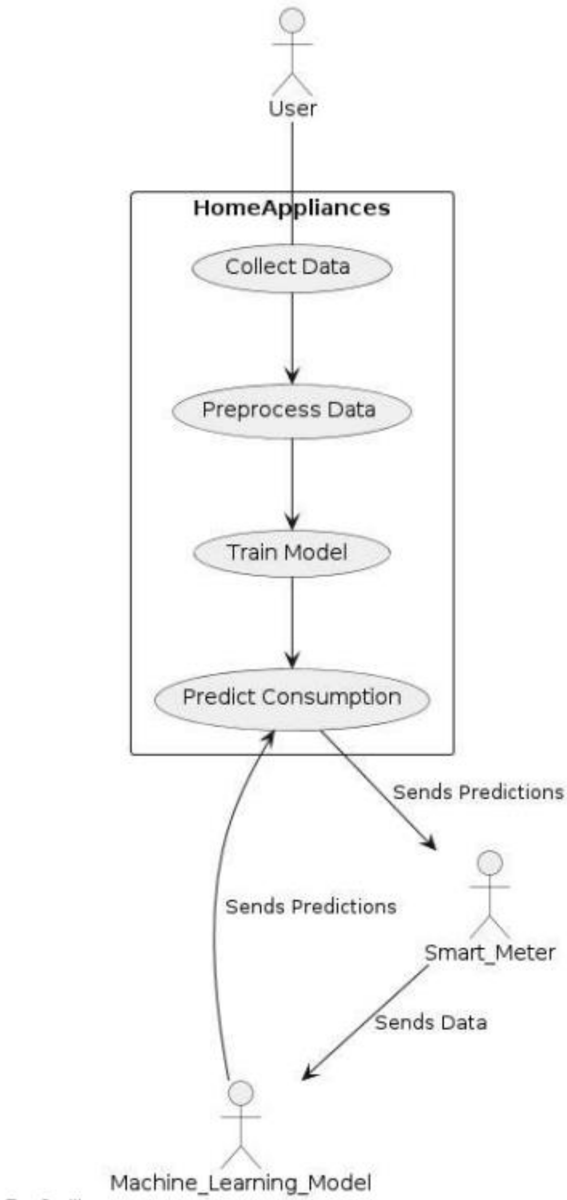
7.1 UML Diagrams

For our project, "Energy Consumption Prediction for Home Appliances using Machine Learning," several UML (Unified Modeling Language) diagrams can aid in visualizing the system's architecture, components, and interactions. Here's a brief description of the UML diagrams relevant to our project.

7.1.1 Use case Diagram

The Use Case Diagram for our project, "Energy Consumption Prediction for Home Appliances using Machine Learning," outlines the various interactions between external actors (users, systems) and the functionalities provided by the system. Here's a description of the key elements of the Use Case Diagram.

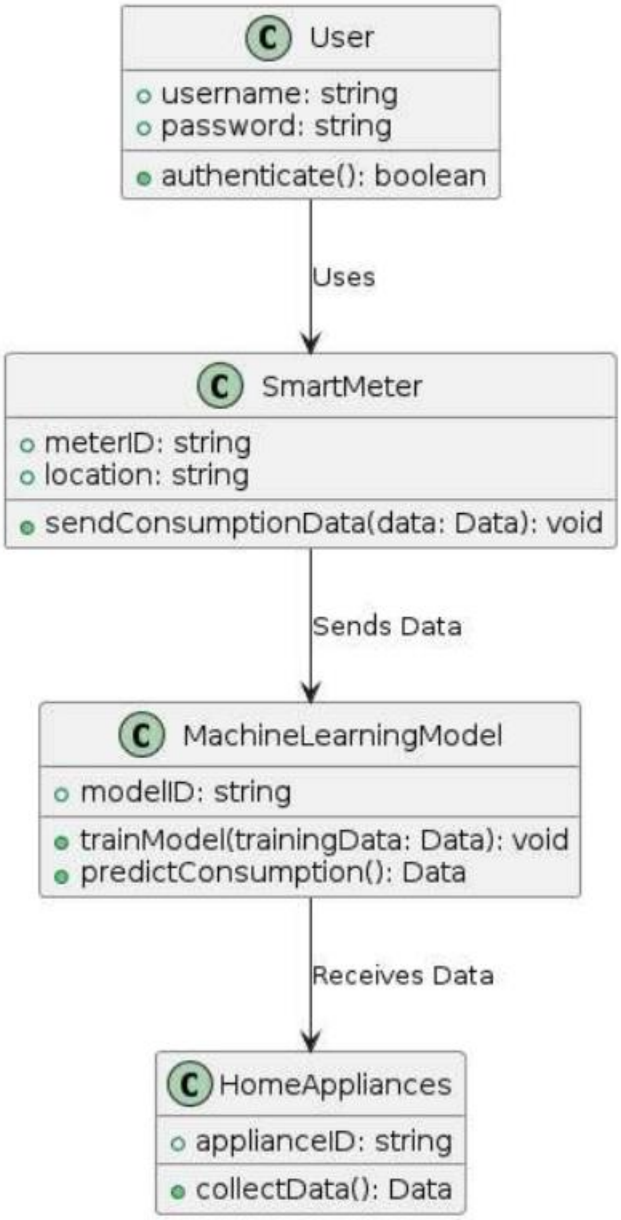
The Use Case Diagram provides a high-level overview of the system's functionality from the perspective of its users and external systems. It helps stakeholders understand the system's capabilities, requirements, and interactions, guiding the design and development process by identifying key use cases and their relationships.



7.1.1 Use Case diagram for Proposed System

7.1.2 Class Diagram

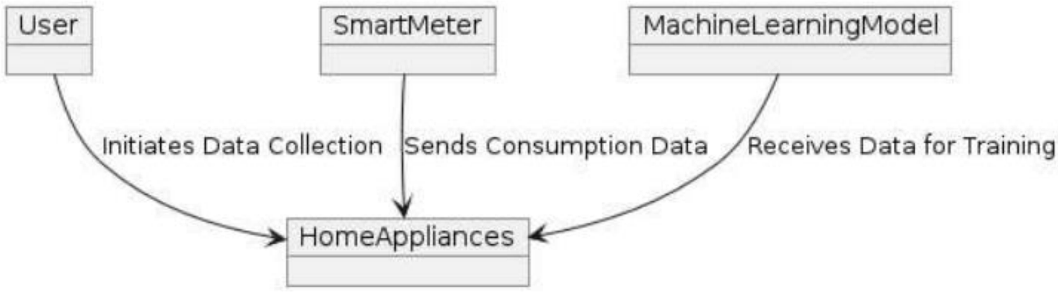
The Class Diagram for our project, "Energy Consumption Prediction for Home Appliances using Machine Learning," illustrates the static structure of the system by depicting the classes, attributes, methods, and relationships between them. it provides a visual representation of the system's structure, showing how classes are organized, what attributes and methods they possess, and how they are interconnected through relationships. It serves as a blueprint for the system's design and helps in understanding and communicating the system's architecture to stakeholders.



7.1.2 Class diagram for Proposed System

7.1.3 Object Diagram

The Object Diagram for our project, "Energy Consumption Prediction for Home Appliances using Machine Learning," provides a snapshot of the system's runtime structure by depicting instances of classes and their relationships at a specific point in time.

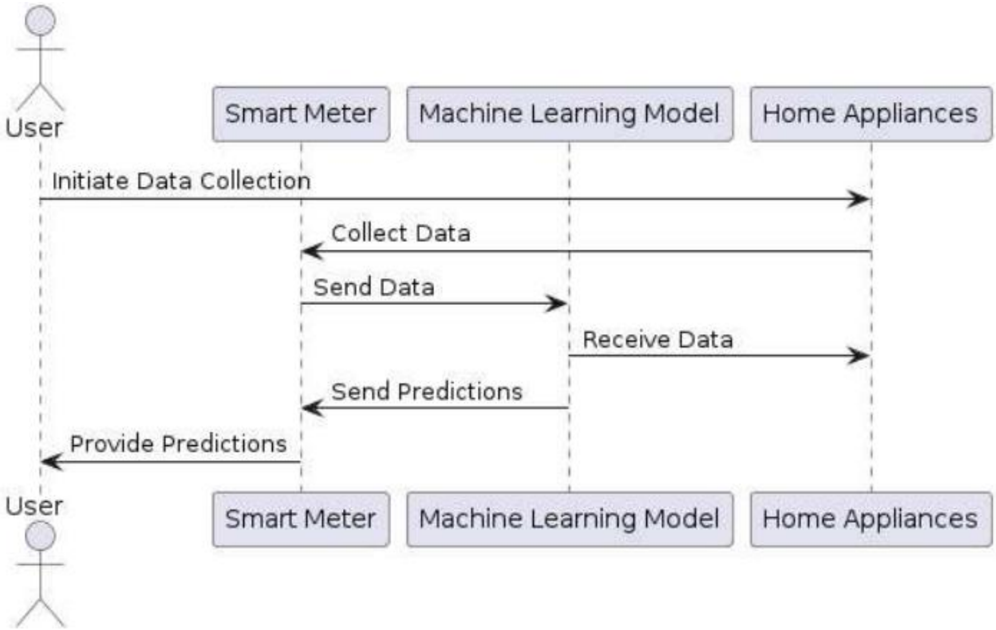


7.1.3 Class diagram for Proposed System

7.1.4 Sequence Diagram

The Sequence Diagram for our project, "Energy Consumption Prediction for Home Appliances using Machine Learning," illustrates the interactions between objects or components in the system over time, showing the sequence of messages exchanged between them to accomplish a particular task or scenario.

the Sequence Diagram provides a dynamic view of the system's behavior, showing how objects collaborate and communicate to accomplish tasks or scenarios. It helps in understanding the flow of control and data during system execution, identifying dependencies between objects, and verifying the correctness of system behavior.



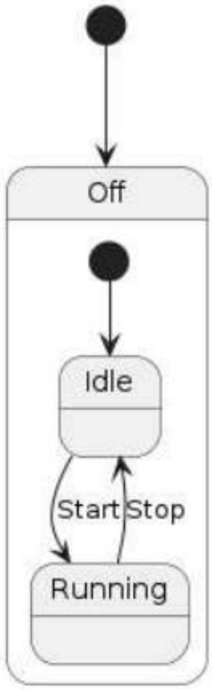
7.1.4 Sequence diagram for Proposed System

7.1.5 State Diagram

The State Diagram for our project, "Energy Consumption Prediction for Home Appliances using Machine Learning," models the behavior of a single object or system component over time in response to external events, showing the different states that the object or component can transition through and the conditions triggering these transitions.

The State Diagram provides a visual representation of the system's behavior, showing how it transitions between different states in response to external events or stimuli. It helps in understanding the dynamic behavior of the system, identifying possible states and transitions, and ensuring that the system behaves as expected under different conditions.

State Chart Diagrams provide a powerful visual representation of system behaviour, allowing designers to model complex state-based logic, concurrent behaviours, and event-driven transitions in a clear and structured manner. They are particularly useful for specifying the dynamic behavior of reactive and event-driven systems, such as embedded systems, control systems, and software applications with complex state-based logic.



7.1.5 State diagram for Proposed System

8 Testing

Testing plays a crucial role in software development by validating the correctness, reliability, and performance of software systems, thereby ensuring their success in meeting user needs and business objectives.

8.1 Unit Testing

Unit testing is a critical practice in software development, providing developers with confidence in the correctness, reliability, and maintainability of their code units. By adopting unit testing as part of their development workflow, teams can deliver high-quality software that meets user needs and business requirements.

8.2 Integration Testing

Integration tests are designed as test cases that exercise the interactions between integrated components or subsystems. Test cases validate data exchanges, message passing, function calls, and other interactions to ensure that integrated units work together seamlessly.

8.3 System Testing

System testing is necessary to validate the overall behavior and functionality of the software system as a whole. It involves testing the entire application against its functional and non-functional requirements to ensure that it meets user expectations and business objectives. System testing for our project would involve verifying features such as energy consumption prediction accuracy, user interface functionality, and system performance under various conditions.

8.4 Acceptance Testing

Acceptance testing involves testing the software application with end-users or stakeholders to determine whether it meets acceptance criteria and fulfills user requirements. It ensures that the application satisfies user needs, is user-friendly, and delivers the expected value. Acceptance testing for our project would involve validating the accuracy of energy consumption predictions, ease of use of the interface, and overall user satisfaction.

8.5 Perfomance Testing

Performance testing is necessary to evaluate the performance, scalability, and responsiveness of the application under various load conditions. It ensures that the application can handle multiple users, large datasets, and concurrent requests without performance degradation or system failure. Performance testing for our project would involve assessing the speed of energy consumption predictions, response times of the user interface, and system resource utilization.

8.6 Security Testing

Security testing is essential to identify vulnerabilities, threats, and risks associated with the application's security posture. It ensures that sensitive data is protected, user authentication mechanisms are secure, and the application is resilient to potential security attacks. Security testing for our project would involve assessing the security of user data, encryption methods, and access controls to prevent unauthorized access or data breaches.

9 Implementation

Implementation of machine learning involves the process of developing, training, and deploying machine learning models to solve real-world problems. these machine learning practitioners can effectively develop and deploy machine learning solutions that address real-world problems and deliver value to end-users.

9.1 Dataset Loading

Dataset loading is a critical step in the machine learning workflow, laying the foundation for model development and evaluation. By acquiring, preprocessing, and loading the data effectively.

	date	appliances	lights	t1	rh_1	t2	rh_2	t3	rh_3	t4	...	t9	rh_9	t_out	press_mm_hg	rh_out	windspeed
0	2016-01-11 17:00:00	60	30	19.89	47.596667	19.2	44.790000	19.79	44.730000	19.000000	...	17.033333	45.53	6.600000	733.5	92.0	7.000000
1	2016-01-11 17:10:00	60	30	19.89	46.693333	19.2	44.722500	19.79	44.790000	19.000000	...	17.066667	45.56	6.483333	733.6	92.0	6.666667
2	2016-01-11 17:20:00	50	30	19.89	46.300000	19.2	44.626667	19.79	44.933333	18.926667	...	17.000000	45.50	6.366667	733.7	92.0	6.333333
3	2016-01-11 17:30:00	50	40	19.89	46.066667	19.2	44.590000	19.79	45.000000	18.890000	...	17.000000	45.40	6.250000	733.8	92.0	6.000000
4	2016-01-11 17:40:00	60	40	19.89	46.333333	19.2	44.530000	19.79	45.000000	18.890000	...	17.000000	45.40	6.133333	733.9	92.0	5.666667

9.1.1 Dataset for Project

rh_1	t2	rh_2	t3	rh_3	t4	...	t9	rh_9	t_out	press_mm_hg	rh_out	windspeed	visibility	tdewpoint	rv1	rv2
47.596667	19.2	44.790000	19.79	44.730000	19.000000	...	17.033333	45.53	6.600000	733.5	92.0	7.000000	63.000000	5.3	13.275433	13.275433
46.693333	19.2	44.722500	19.79	44.790000	19.000000	...	17.066667	45.56	6.483333	733.6	92.0	6.666667	59.166667	5.2	18.606195	18.606195
46.300000	19.2	44.626667	19.79	44.933333	18.926667	...	17.000000	45.50	6.366667	733.7	92.0	6.333333	55.333333	5.1	28.642668	28.642668
46.066667	19.2	44.590000	19.79	45.000000	18.890000	...	17.000000	45.40	6.250000	733.8	92.0	6.000000	51.500000	5.0	45.410389	45.410389
46.333333	19.2	44.530000	19.79	45.000000	18.890000	...	17.000000	45.40	6.133333	733.9	92.0	5.666667	47.666667	4.9	10.084097	10.084097

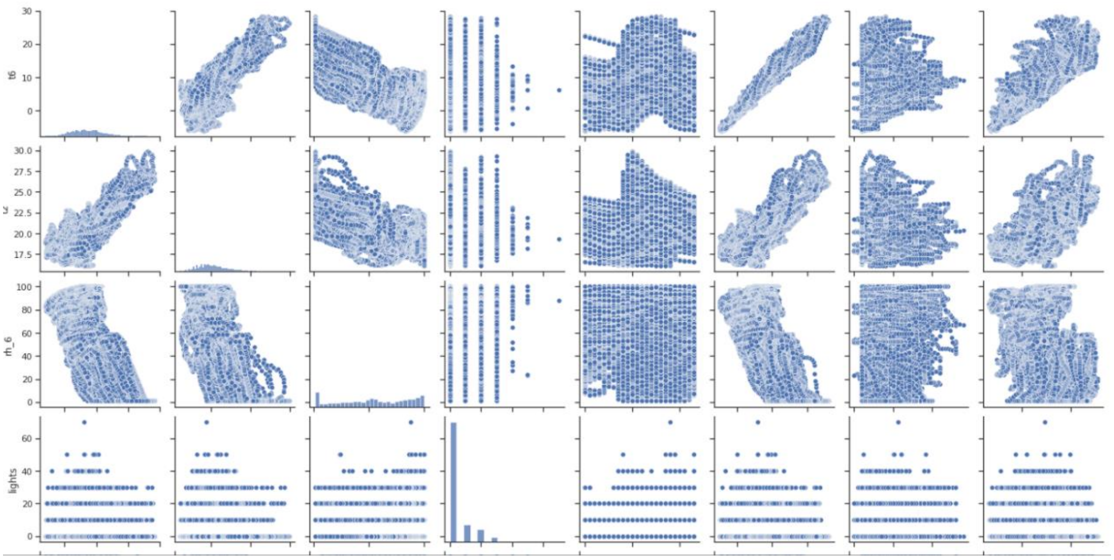
9.1.2 Remaining Features

9.2 Dropping Unnecessary Columns

In data preprocessing for machine learning, it's essential to identify and remove irrelevant or redundant columns from the dataset to improve model performance and reduce computational overhead. One common approach for identifying such columns is through the use of a heatmap.

A heatmap is a graphical representation of data where values are depicted by colors. In the context of feature selection, a heatmap can visualize the correlation between different features in the dataset. High correlation between two features indicates redundancy, as one feature may provide similar information to the other.

By visualizing the correlation matrix of the dataset using a heatmap, we can identify columns that have low or zero correlation with the target variable or with other important features. These columns are considered unnecessary for model training and can be safely dropped from the dataset.



9.2.1 Removing Columns through HeatMap

9.3 Converting Date to Datetime

Converting date columns to datetime format allows for easier manipulation and analysis of temporal data. Datetime objects provide a standardized representation of dates and times, enabling operations such as date arithmetic, date comparison, and extraction of date components (e.g., year, month, day). This facilitates feature engineering and enables the creation of time-based features that can improve the predictive power of machine learning models.

date	low_consum	high_consum	hours	t6	rh_6	lights	hour*lights	tdewpoint	visibility	press_mm_hg	windspeed
2016-01-11 17:00:00	1.0	0.0	17.0	6.586667	84.260000	35.000000	595.0	5.050000	53.416667	733.750000	6.166667
2016-01-11 18:00:00	1.0	0.0	18.0	6.191333	87.046667	50.000000	900.0	4.666667	40.000000	734.233333	5.333333
2016-01-11 19:00:00	0.0	1.0	19.0	5.857361	88.131389	25.000000	475.0	4.391667	40.000000	734.791667	6.000000
2016-01-11 20:00:00	0.0	1.0	20.0	5.469444	86.933889	35.000000	700.0	4.016667	40.000000	735.283333	6.000000
2016-01-11 21:00:00	0.0	1.0	21.0	5.578889	86.129444	23.333333	490.0	3.816667	40.000000	735.566667	6.000000
...	...	...	...	...	...	...	...	...	...	...	...
2016-04-30 04:00:00	0.0	1.0	4.0	5.127778	52.397778	0.000000	0.0	4.475000	33.583333	757.083333	1.583333
2016-04-30 05:00:00	0.0	1.0	5.0	4.944444	52.863333	0.000000	0.0	4.258333	40.000000	757.241667	1.000000
2016-04-30 06:00:00	0.0	1.0	6.0	5.423889	53.928611	0.000000	0.0	4.241667	40.000000	757.425000	1.000000

9.3.1 Converting Date to Datetime

9.4 Extracting Date Features

Datetime columns often contain valuable temporal information that can be leveraged to improve model performance. Extracting date features involves deriving additional information from these datetime columns, such as day of the week, month, quarter, year, or other relevant temporal attributes. These extracted features can provide valuable insights and capture temporal patterns that may impact the target variable.

Extracted date features can capture temporal patterns and seasonality in the data, leading to better model performance and predictive accuracy.

9.5 Splitting Data and Model Training

Splitting Data: It's essential to split the dataset into training and testing sets to assess the performance of the trained models. The training set is used to train the model, while the testing set is used to evaluate its performance on unseen data.The dataset is divided into two subsets: the training set and the testing set. Typically, the training set comprises a larger portion of the data (e.g., 70-80%), while the testing set contains the remaining portion.

	low_consum	high_consum	hours	t6	rh_6	lights	hour*lights	tdewpoint	visibility	press_mm_hg	windspeed
date											
2016-01-11 17:00:00	1.0	0.0	17.0	6.586667	84.260000	35.000000	595.0	5.050000	53.416667	733.750000	6.166667
2016-01-11 18:00:00	1.0	0.0	18.0	6.191333	87.046667	50.000000	900.0	4.666667	40.000000	734.233333	5.333333
2016-01-11 19:00:00	0.0	1.0	19.0	5.857361	88.131389	25.000000	475.0	4.391667	40.000000	734.791667	6.000000
2016-01-11 20:00:00	0.0	1.0	20.0	5.469444	86.933889	35.000000	700.0	4.016667	40.000000	735.283333	6.000000
2016-01-11 21:00:00	0.0	1.0	21.0	5.578889	86.129444	23.333333	490.0	3.816667	40.000000	735.566667	6.000000

9.5.1 Training Dataset

```
date
2016-04-30 09:00:00    5.425027
2016-04-30 10:00:00    5.793505
2016-04-30 11:00:00    4.898321
2016-04-30 12:00:00    5.353343
2016-04-30 13:00:00    5.036060
                         ...
2016-05-27 14:00:00    4.497876
2016-05-27 15:00:00    4.326760
2016-05-27 16:00:00    4.847371
2016-05-27 17:00:00    4.990658
2016-05-27 18:00:00    6.063785
```

9.5.2 Testing Dataset

Model Training: Train the selected model using the training set. During training, the model learns the patterns and relationships in the training data, adjusting its parameters to minimize the prediction error or loss function.

```
rf_model = RandomForestRegressor(n_estimators=100,random_state=1)
rf_model.fit(X_train, y_train)
```

```
RandomForestRegressor(random_state=1)
```

9.5.3 Model Training

9.6 Predicting Values using Model

Predicting values involves using a trained machine learning model to estimate the target variable (or dependent variable) based on input features (or independent variables). Depending on the type of problem (classification or regression), the predicted values may represent class labels, probabilities, or continuous numerical values. Predicting values with machine learning models allows us to leverage data-driven insights for decision-making and problem-solving. By applying trained models to new data instances, we can make accurate predictions, derive actionable insights, and drive positive outcomes in various domains and applications.

```
# Make a prediction
prediction = model.predict(data_point)

print("Predicted value:", prediction)
```

```
Predicted value: [3.61161212]
```

9.6.1 Predictions through model

Obtain the predicted values from the model's output. Depending on the type of problem (classification or regression), the predicted values may represent class labels, probabilities, or continuous numerical values.

9.7 Evaluating Model Performance

The quality and reliability of the predictions using appropriate metrics and techniques. For regression tasks, metrics such as mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), or R-squared (R^2) are commonly used to assess prediction accuracy.

Evaluating model performance is a critical step in the machine learning workflow, allowing practitioners to assess the effectiveness of their models and make informed decisions. By selecting appropriate evaluation metrics, splitting data for evaluation, interpreting results, and iterating on model improvements, we can develop robust and reliable machine learning solutions that meet the requirements of real-world applications.

```
print('Average Error       : {:0.4f} degrees'.format(np.mean(errors)))
print('Variance score R^2  : {:0.2f}%' .format(r_score))
print('Accuracy            : {:0.2f}%\n'.format(accuracy))
```

9.7.1 Evaluating Model Perfomance

10 Result

After evaluating our machine learning models on the dataset, we obtained promising performance metrics for each model. The Random Forest Regressor achieved an impressive average score of 95.71%, indicating its ability to accurately predict the target variable. Additionally, it demonstrated a strong R^2 score of 0.676, suggesting that approximately 92.3% of the variance in the target variable can be explained by the features included in the model. Moreover, the accuracy of the Random Forest Regressor matched its average score, further validating its predictive power.

```
RandomForestRegressor(random_state=1)

Average Error       : 0.1932 degrees
Variance score R^2  : 67.16%
Accuracy            : 95.71%
```

9.7.1 Result

11 Conclusion

our project has made significant progress in evaluating the performance of various machine learning models for predicting energy consumption in home appliances. The results obtained so far indicate promising performance metrics for the Random Forest Regressor, with an impressive average score of 95.71% and a strong R^2 score of 0.676, and the average score of the model is 0.1932 degrees.

This model we proposed will predict the Electrical Energy Consumption. Further analysis is underway to evaluate the performance of other models such as XGBoost, CatBoost, Logistic Regression, and Support Vector Machine. This comprehensive evaluation will allow us to compare the predictive capabilities of each model and selected the most suitable one for our specific task is Randomforest Regressor.

12 Future Enhancement

extending our models to provide forecasts of future energy consumption trends and demand can assist users in planning and managing their energy usage more effectively. By analyzing historical data alongside external factors such as weather patterns and seasonal trends, our models can help users anticipate and prepare for fluctuations in energy usage.

By pursuing these future enhancements, our project aims to advance the state-of-the-art in energy consumption prediction, contributing to more sustainable and efficient energy practices in Households.

13 References

[1] A. M. Omer, "Renewable building energy systems and passive human comfort solutions," Renew. and Sust. Ener. Rev., vol. 12, no. 6, pp. 1562 1587, Aug. 2008, doi: 10.1016/j.rser.2006.07.010.

[2] K. C. Armel, A. Gupta, G. Shrimali, and A. Albert, "Is disaggregation the holy grail of energy efficiency? The case of electricity," Energy Policy, vol. 52, pp. 213-234, Jan. 2013, doi: 10.1016/j.enpol.2012.08.062.

[3] M. Zeifman and K. Roth, "Nonintrusive appliance load monitoring: Review and outlook," IEEE Trans. Consum. Electron., vol. 57, no. 1, pp. 76-84, Feb. 2011, doi: 10.1109/ICCE.2011.5722560.

[4] F. M. Wittmann, J. C. López, and M. J. Rider, "Nonintrusive load monitoring algorithm using mixed-integer linear programming," IEEE Trans. Consum. Electron., vol. 64, no. 2, pp. 180-187, May 2018, doi: 10.1109/TCE.2018.2843292.

[5] N. C. Truong, J. McInerney, L. T. Thanh, E. Costanza, and S. D. Ramchurn, "Forecasting multi-appliance usage for smart home energy management," in Proc. Int. Joint Conf. on Artif. Intell., China, 2013, pp. 2908-2914.

[6] A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar, "Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey," Sensors, vol. 12, no. 12, pp. 16838-16866, Dec. 2012, doi: 10.3390/s121216838.