Demand-Side Electricity Management through Machine Learning

A PROJECT REPORT

Submitted in partial fulfilment for the award of the degree of

Bachelor of Engineering

IN

Artificial Intelligence and Machine Learning

Submitted by:

Rishabh Joshi, 20BCS6869

Rahul Kumar, 20BCS6861

Under the Supervision of:

PRABJOT SINGH BALI

CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413, PUNJAB

MAY 2024

BONAFIDE CERTIFICATE

Certified that this project report **"DEMAND SIDE ELECTRICITY MANANGEMENT THROUGH MACHINE LEARNING"** is the bonafide work of **RISHABH JOSHI AND RAHUL KUMAR** who carried out the project work under my/our supervision.

SIGNATURE	SIGNATURE
Dr. Aman Kaushik	prof. Prabjot Singh Bali
HEAD OF THE DEPARTMENT	SUPERVISOR

Artificial Intelligence and Machine Learning

Submitted for the project viva-voce examination held on 30 April 2024

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

DECLARATION

I, **Rahul , Rishabh joshi** students of 'Bachelors of engineering in CSE (HONS.) with specialization in artificial intelligence and machine learning in association with IBM', session: 2020-24, Department of Computer Science and Engineering, Apex Institute of Technology, Chandigarh University, Punjab, hereby declare that the work presented in this Project Work entitled **'Demand-Side Electricity Management through Machine Learning'** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

ACKNOWLEDGMENT

It is indeed with great pleasure and an immense sense of gratitude that we acknowledge the help of these individuals. We are highly indebted to our Chancellor **S. SATNAM SINGH SANDHU, VICE- CHANCELLOR Dr. H.B. RAGHAVENDAR, CHANDIGARH UNIVERSITY**, for the facilities provided to accomplish this main project. We would like to thank our **PRABJOT SINGH BALI SIR**, for this constructive criticism throughout our project. We feel elated in manifesting our sense of gratitude to our internal project supervisor PROF **PRABJOT SINGH BALI** , He has been a constant source of inspiration for us and we are very deeply thankful to him for his support and valuable advice. We are extremely grateful to our Departmental staff members for their extreme help throughout our project. Finally, we express our thanks to all of our friends who helped us in the successful completion of this project.

Project Associates:

RAHUL KUMAR (20BCS6861)

RISHABH JOSHI (20BCS6869)

TABLE OF CONTENTS

List of Figures

List of Tables

ABSTRACT

Rising energy costs and the increasing integration of renewable energy sources necessitate advancements in household electricity management. This project explores the potential of machine learning to optimize energy usage within residential settings.

We investigate the development of a machine learning model capable of predicting household electricity consumption. The project focuses on analysing historical data on total energy usage spanning a three-year period. By examining this data, we aim to identify patterns and trends that influence electricity consumption, including seasonal variations and potential correlations with external factors.

Through a process of feature engineering and selection, we identify the most relevant features for model development. Considering constraints like data availability, privacy, and model interpretability, we ensure the resulting model is not only accurate but also practical and user-friendly.

The project follows a well-defined design flow, encompassing data collection and preprocessing, feature engineering and selection, model training and evaluation, and finally, deployment and monitoring. By implementing this machine learning approach, we aim to empower households to manage their energy use more efficiently, ultimately contributing to a more sustainable and cost-effective energy landscape.

We aim to predict the energy consumption in household. More specifically we try to predict the electricity consumption in a household to get better insights of the data. The primary objective is to learn the patterns in the electricity consumption and further test the accuracy of our result. The project makes use of signal processing methods and some statistical methods to better understand the electricity usage patterns and predict the usage of the energy in future. There are many techniques in signal processing and machine learning that, when combined are a very good analyser for the given system. Here, in this project we do the analysis using the Fourier transform and convolution to get to the results. We also make use of convolution and curve fitting to best analyse our result.

CHAPTER 1.

INTRODUCTION

1.1 Machine Learning

Machine learning (ML) is a powerful subfield of artificial intelligence (AI) that empowers computers to learn from data without explicit programming. Unlike traditional software that relies on pre-defined rules, ML algorithms are like students constantly studying and improving based on the information they're exposed to.

Imagine a vast library filled with books on various topics. Traditional software would be like a single, pre-written instruction manual, limited to a specific task. Machine learning, on the other hand, is like a curious researcher who reads and analyzes all the books, identifying patterns, relationships, and hidden knowledge within the data.

This ability to learn and adapt is what makes ML so versatile. In the context of electricity consumption, ML algorithms can be trained on massive datasets containing information on past energy usage, weather patterns, appliance types, and even occupancy levels in homes. By analyzing these vast troves of data, ML models can uncover subtle patterns and relationships that influence how much energy a household uses.

Once these patterns are identified, the real magic happens. ML models can then use their newfound knowledge to make predictions about future electricity consumption. They can estimate how much energy a household will likely use on a given day, factoring in seasonal variations and even external influences like weather forecasts. This predictive power is what unlocks the potential for optimizing energy management in homes and beyond.

Machine learning, a branch of artificial intelligence, revolves around creating algorithms and statistical models that enable computers to perform tasks without explicit programming. Instead of relying on predefined rules, these algorithms learn from data patterns and relationships, facilitating predictions or decisions.

Machine learning encompasses different approaches:

Supervised Learning: This method involves training a model on labeled data, where each input corresponds to a known output. Through this process, the model learns to associate inputs with correct outputs, enabling it to make predictions on new data.

Unsupervised Learning: Here, the model is presented with unlabeled data and tasked with identifying inherent patterns or structures within it. Common tasks include clustering and dimensionality reduction.

Semi-Supervised Learning: Combining aspects of both supervised and unsupervised learning, this approach utilizes a mix of labeled and unlabeled data for training, particularly useful when labeled data is limited or costly.

Reinforcement Learning: This learning paradigm involves an agent learning optimal decision-making strategies by interacting with an environment and receiving feedback in the form of rewards or penalties based on its actions.

Machine learning finds applications across various domains, including image and speech recognition, natural language processing, recommendation systems, autonomous vehicles, finance, and healthcare. Recent advancements, fueled by the availability of vast datasets and computational resources, have significantly propelled progress in AI-related fields.

1.2 Fourier Transform

The Fourier Transform (FT) is a powerful mathematical tool used to analyze signals, often those that appear complex at first glance. Imagine a sound wave – like the roar of a crowd at a sporting event. It might seem like a chaotic jumble of noise, but the FT can break it down into its fundamental building blocks: frequencies.Think of a sound wave as a musical composition. Each instrument plays at a specific frequency, creating its unique tone. The FT acts like a conductor, revealing the individual frequencies present in the overall sound and their relative strengths. It tells us how much each frequency contributes to the overall sound wave, similar to a spectrum analyzer showing the intensity of different colors in a beam of light.

Here's where things get interesting. Once we understand the frequencies that make up a signal, we can manipulate it in fascinating ways. Imagine wanting to adjust the volume of a specific instrument in a song. The FT allows us to perform a similar feat with signals. We can selectively

filter out unwanted frequencies, like removing the background hum from an old vinyl recording.

However, the FT isn't just about sound waves. It's a versatile tool used across various scientific and engineering fields. For instance, an FT can analyze the vibrations of a bridge to identify potential structural weaknesses. In image processing, it can be used to sharpen blurry pictures or remove noise from medical scans.

The key concept lies in the relationship between the original signal (like the sound wave) and its frequency domain representation (the individual frequencies). The FT essentially translates a signal from the time domain (how it changes over time) to the frequency domain (what frequencies are present). This allows us to analyze and manipulate signals in ways that wouldn't be possible by simply looking at their raw form.

While the mathematical underpinnings of the FT can be complex, the core idea is quite intuitive. By understanding the hidden frequencies that make up a signal, the Fourier Transform unlocks a powerful way to analyze, manipulate, and ultimately, understand the world around us in a whole new light.

The Fourier Transform is a mathematical method devised by Joseph Fourier that breaks down a function into its constituent frequencies. It's crucial in analyzing both continuous and discrete signals.

1. **Continuous Fourier Transform (CFT):** This deals with continuous signals, converting them from the time (or space) domain to the frequency domain.

2. **Discrete Fourier Transform (DFT):** This is for discrete, sampled signals, transforming sequences of numbers into frequency components.

The Fast Fourier Transform (FFT) algorithm efficiently computes the DFT, finding widespread use in digital signal and image processing, data compression, and scientific research.

Applications of the Fourier Transform include signal processing (such as telecommunications and audio), image processing, spectral analysis, quantum mechanics, and data compression.

In essence, the Fourier Transform is a foundational tool across mathematics and engineering, vital for understanding and manipulating signals and data in diverse applications.

1.3 Problem Identification

The efficient management of electricity is crucial for a stable and sustainable energy future. However, balancing supply and demand on the consumer side (homes and businesses) presents several significant challenges. Here's a closer look at some key problems:

1. **Peak Demand and Grid Strain**: Electricity demand isn't constant throughout the day. Certain times, like hot summer afternoons when everyone uses air conditioning, experience a surge in demand, creating "peaks." Power grids need to be able to handle these peaks, but relying solely on building new power plants is expensive and environmentally unfriendly.

2. **Unpredictable Consumption Patterns:** Unlike water usage, which tends to be relatively consistent, electricity consumption varies significantly. Factors like weather (heating in winter, cooling in summer), occupant behavior (working from home), and appliance usage can lead to unpredictable fluctuations. This variability makes it difficult for utilities to accurately forecast demand and allocate resources efficiently.

3. **Limited Visibility into User Behavior:** Traditional electricity meters often only provide total consumption data, offering little insight into how specific appliances or activities contribute to overall usage. This lack of granular information makes it challenging for consumers to identify areas for potential energy savings or adjust their habits for optimal efficiency.

4. **Lack of User Engagement and Incentives:** Consumers often lack awareness or motivation to actively manage their electricity use. Without clear information on usage patterns and cost implications, it can be difficult to understand the impact of individual

choices. Additionally, the absence of effective incentives can limit consumer adoption of energy-saving practices and technologies.

5. **Technological Integration and Infrastructure:** Optimizing electricity usage often requires integrating smart metering technologies with user interfaces and automated control systems. However, this can be hindered by the lack of standardized technologies, compatibility issues across different devices, and the cost of implementing such infrastructure on a large scale.

These challenges highlight the need for innovative solutions that promote greater user engagement, leverage data analytics for informed decision-making, and utilize technology to optimize electricity consumption on the demand side. This will pave the way for a more sustainable and efficient electricity grid that benefits both consumers and utilities.

- **Growing Electricity Demand:**
- **Peak Demand Challenges:**
- **Limited Awareness of Optimal Consumption Times:**
- **Inadequate Tools for Consumer Energy Management:**
- **Environmental Consequences of Peak Demand:**
- **Lack of Dynamic Adaptability in Energy Systems:**

Certainly! Here's a explaination:

1. Growing Electricity Demand: With population growth and technological advancements, the need for electricity continues to escalate, straining existing infrastructure and necessitating innovative management strategies.

2. Peak Demand Challenges: During peak demand periods, typically occurring at specific times, the electricity grid faces increased pressure, leading to higher costs and reliability concerns. Utilizing machine learning can help predict and manage these peaks, easing strain on the grid.

3.Limited Awareness of Optimal Consumption Times: Many consumers lack awareness of when energy demand peaks and electricity costs are highest. Machine learning algorithms can analyze data to identify optimal consumption times, empowering consumers to make informed decisions about energy usage.

4.Inadequate Tools for Consumer Energy Management: Traditional energy management tools often lack sophistication and fail to provide personalized insights. Machine learning offers more advanced solutions, such as predicting consumption patterns and offering tailored energy-saving strategies.

5. Environmental Consequences of Peak Demand: Meeting peak demand often requires increased use of fossil fuels, leading to higher greenhouse gas emissions and environmental harm. By effectively managing peak demand with machine learning, reliance on polluting energy sources can be reduced, mitigating environmental impacts.

6. Lack of Dynamic Adaptability in Energy Systems: Traditional energy systems struggle to adapt to changing demand and external factors. Machine learning enables systems to become more dynamic, predicting demand fluctuations in real-time and integrating renewable energy sources more effectively, enhancing sustainability and resilience.

1.4 Identification of Tasks

In order to build a solution to a problem, tasks need to be identified. Only after the task identification can we proceed further in the investigation of the problem. The tasks for electricity prediction can be identified into the following parts:

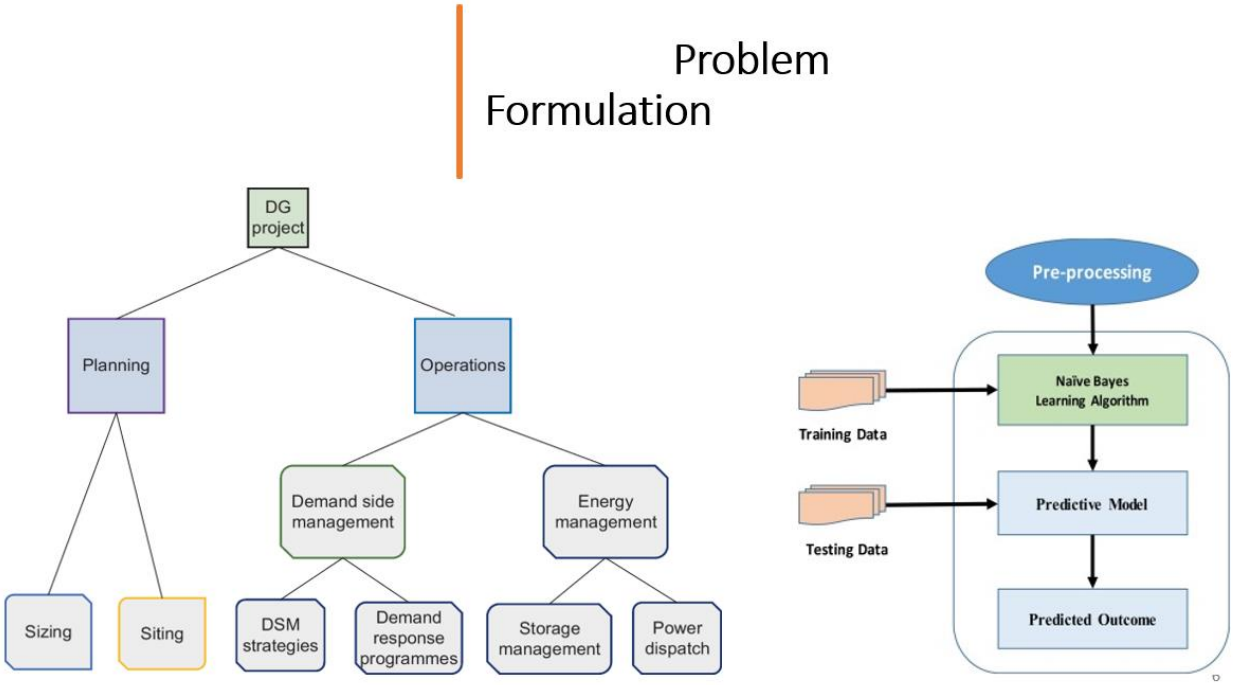
Consumption Pattern Recognition: The first task involves analysing historical data on electricity usage to identify patterns and trends. This includes understanding how consumption varies by season, time of day, weather conditions, and occupant behaviour (in homes and buildings). Machine learning algorithms can be trained on this data to identify anomalies, predict future usage patterns, and classify different types of energy-consuming activities.

Demand Forecasting and Optimization: With a clear understanding of consumption patterns, machine learning can then be used for demand forecasting. This involves predicting future electricity needs based on historical data, weather forecasts, and real-time information like grid conditions. By accurately predicting demand, utilities can optimize their generation and distribution strategies, minimizing reliance on expensive peak power plants and ensuring a stable electricity supply.

1.5 Organization of the Report

This report delves into the potential of machine learning for optimizing electricity consumption within households. It presents a comprehensive investigation that explores the design flow, methodology, and results achieved in developing a machine learning model for electricity usage prediction.

1.5.1 Introduction



The introductory chapter sets the stage by highlighting the growing importance of efficient electricity management in homes. It discusses the rising energy demands, the increasing integration of renewable energy sources, and the need for innovative solutions. This chapter introduces machine learning as a powerful tool for optimizing household electricity usage and outlines the objectives of the research project.

1.5.2 Design Flow

Chapter 2 provides a roadmap for the project, outlining the well-defined design flow followed throughout the development process. It details the key stages involved, including:

1.5.2.1 **Data Collection and Preprocessing**: This section describes the methods used to gather data on household electricity usage. It could involve utilizing smart meter data, utility bills, or surveys. The chapter then discusses the steps taken to clean and organize the collected data, ensuring accuracy and consistency for further analysis.

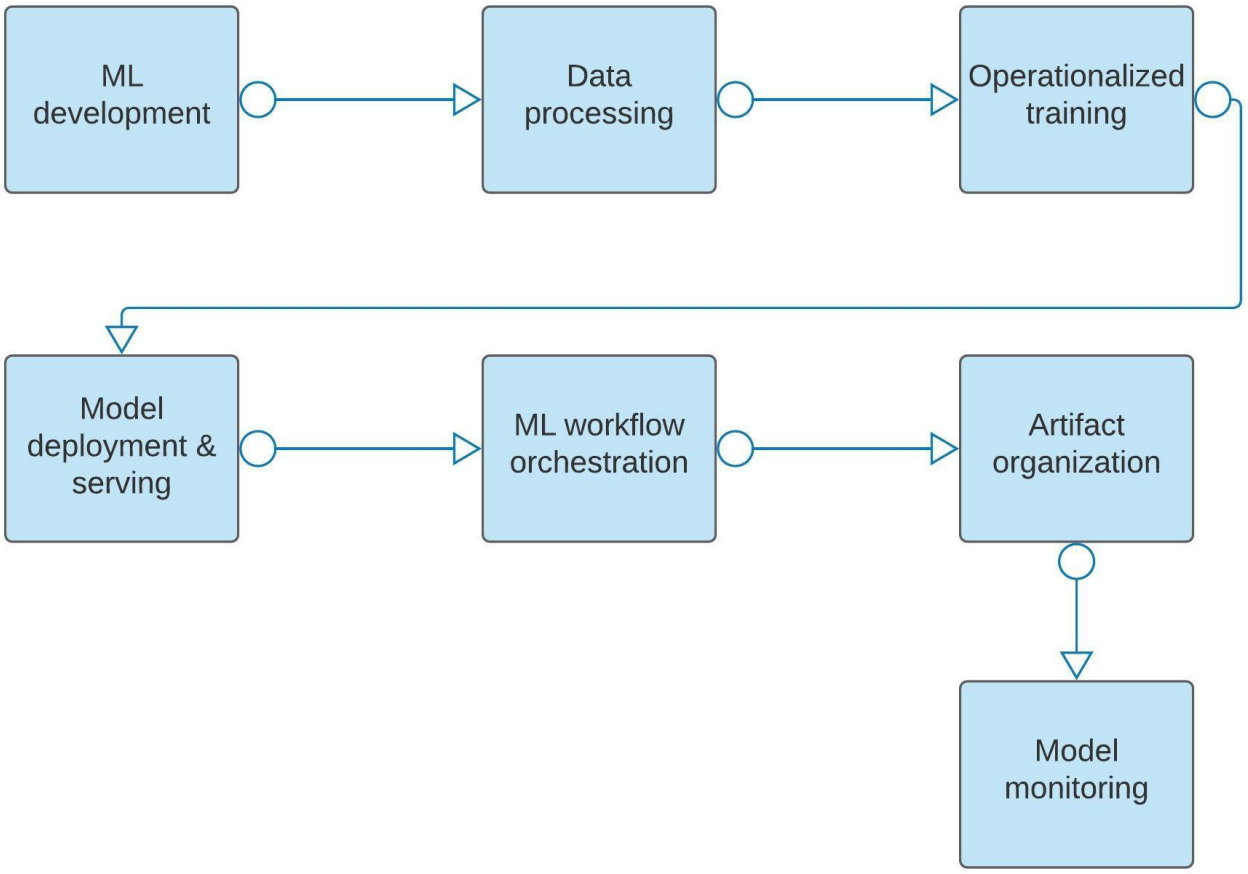
1.5.2.2 **Feature Engineering and Selection:** This section delves into the process of identifying the most impactful features that influence electricity consumption. It explains how the data was analyzed to uncover patterns, trends, and potential correlations with external factors like weather. The chapter also details how new features might be created based on existing data (e.g., average daily consumption) through a process called feature engineering. Finally, it describes the selection of the most relevant features for model development, considering constraints like data availability, privacy concerns, and model interpretability.

1.5.2.3 **Model Training and Evaluation:** Chapter 2 outlines the process of building the machine learning model. It describes the selection of a suitable machine learning algorithm and its training on the prepared data. The chapter then explains how the model's performance is evaluated on unseen data to assess its accuracy and generalizability. This section might also discuss iterative refinement of the model or feature selection until a satisfactory level of performance is achieved.

1.5.2.4 **Deployment and Monitoring:** The final stage of the design flow, as detailed in this chapter, focuses on deploying the trained model. It explores options for integrating the model with existing smart home systems or creating a user-friendly interface for households to interact with the model. Finally, the chapter discusses the importance of monitoring the model's performance in real-world use and making adjustments as needed to ensure continued accuracy and value in optimizing electricity consumption.

1.5.3 Methodology

Chapter 3 dives deeper into the specific methodological approaches employed for data analysis, model development, and evaluation. It provides a detailed explanation of the chosen machine learning algorithm, along with the rationale behind its selection. This section might also discuss specific techniques used for data preprocessing, feature engineering, and model training. Additionally, the chapter describes the metrics used to evaluate the model's performance, such as accuracy, precision, and recall.



This architecture involves several steps:

- Data Collection:
- Data Preprocessing:
- Feature Engineering:
- Machine Learning Model Training:
- Model Evaluation:
- Deployment:
- User Interface:
- Monitoring and Maintenance:

1.5.4 Results

The final chapter presents the key findings and results of the research project. It showcases the performance of the developed machine learning model on predicting household electricity consumption. This section might include visualizations or data tables to illustrate the model's prediction accuracy and its ability to capture seasonal variations or weather-related influences. The chapter also discusses any limitations encountered during the project and potential areas for future research or improvement.

1.5.5 Conclusion

By following this structured approach, the report provides a comprehensive overview of the research project, from initial design and methodology to the final results achieved. It offers valuable insights into the feasibility and effectiveness of using machine learning for optimizing electricity consumption within households.

Chapter 2.

LITERATURE REVIEW

It is certain that we are not the first researchers on this topic. We rely heavily on previous researchers and the high-quality paper they put on. They introduced many interesting concepts and applied techniques that helped gain a good understanding of the data. The authors in [1] used four techniques: Bagging, Stochastic Gradient Boosting, Model Averaged Neural Network and the K-Nearest Neighbors. They found out that GBM was able to predict with an outstanding accuracy of 96%. They also argue about the limitations like overfitting, high variance and bias in the data. They concluded that GBM has a great potential and can help accurately predict accuracy of consumption. They ask future researchers to integrate GBM with soft computing methods like Moth Flame Optimization, Harris Hawks Algorithms, etc. The authors in [2] start by telling the importance of predictions and then proceed to categorize the types of prediction in different categories. The authors conclude their research by emphasizing the need of more research in this area. Authors believe that the advancement can lead to savings for the corporations. The authors in [3] make use of Deep learning methods that predict the usage in short term, medium term and long-term. They called the model DLA-PM, Deep Learning Architecture for Power Management. It is designed for managing the energy fluctuations. The home appliances are connected to the cloud server using home automation techniques linked to the servers using IoT system. They conclude by saying that DLA-PM can be used in future years for smart cities to meet the power needs and preserve natural resources in future years. The authors in [4] use ensemble learning techniques like AdaBoost, Bagging, SVM and Decision Tree. The calculation of error made the use of MSE, MAE and Mean Absolute Percentage Error. To reduce the number of features in the dataset, they make the use of PCA and LDA to check the influence of features. They tried to find the optimal number of features for the best prediction. The authors in [5] argue about the increment of the use of ML in this field due to evolving infrastructure. They tried to make the paper more accessible to the people of engineering and CS background because of the lack of real-world models. They argued about the use of physics-based ML approaches that require less data but give good precision. Many papers made the use of the neural networks like, in [6], they compared the ANN and RF for classification purpose. RF means Random Forest. They also modeled the parameters like the number of guests to increase the realism of the prediction and increase the accuracy of their model. For future use, they directed to explore the other methods and big-data technologies for training and deployment. Again, the researchers in [7], did a data-driven predictive modelling of the appliances in our household. They evaluated four statistical methods: Multiple Linear Regression, Support Vector Machine with radial kernel, Random Forest, and Gradient Boosting Machines (GBM), with GBM performing best in all of the test cases taken.

The authors in [7] knew about the limitation of the model. They knew that the data being only for one house cannot be the best in quality for the analysis. The authors in [8], [12], [13] made a model that would predict the next day usage of electricity given today's usage. Their system uses a sensor network tat monitors the consumption to predict next-day usage, this includes what devices will be used, when and how long it will be used. They tried to automate the task as much as possible. The ultimate goal for the authors was to minimize the energy bill and improve smart grid efficiency. The authors in [9], [10] did the work on load forecasting. They argued that the short-term load forecasting is necessary for corporations because it gives idea about the approximate usage of electricity. They also talk about the ANN techniques that are used for the purpose like backpropagation used for calculating the weights in the network. At last, they talk about the limitation of the networks and computing capability of the machines. The authors in [14], did a great job in finding patterns in electricity usage during in the CoVid-19 pandemic. They found that unlike other states like New York which saw a decrease in power usage, Florida and Texas witnessed a surprising increase. Authors in this study propose new machine learning models to improve short-term electricity consumption prediction in the context of COVID- 19's impact. They explored Long-Short-Term-Memory, Convolutional Neural Networks, and Support Vector Machines, comparing performance with ensemble learning techniques. Their work highlights the potential of these during uncertain events. Finally in the paper [14], the authors analyzed the usage of IoT systems in different areas like healthcare and data collection. They also discuss about Wireless Sensor Network or WSN which is a collection of sensors connected to a central point. This paper focused more on the IoT side and the usage in medicine, but it provided a good start by telling the things like WSN, central server and usage in medicine for collecting data.

YEAR	AUTHOR NAME	TITLE OF PROJECT	METHOD USED	RESULTS
FEB 2022	Emmanuel Gbenga Dada	Power Consumption Prediction in Urban Areas using Machine Learning	Gradient Boosting, LDA and other algorithms for finding patterns	Accuracy of 96%
MAR 2013	Dr. Arunesh Kumar Singh, D K Chaturvedi	An Overview of Electricity Demand Forecasting Techniques	genetic algorithms, fuzzy logic, neural networks and knowledge based expert systems etc.	The research has been shifting and replacing old approaches with newer and more efficient ones.
MAR 2021	Mahdi Khodayar	Deep Learning Architecture for Power Management	discriminative, generative, and reinforcement learning approaches	optimize an objective using the observed rewards captured from the problem's environment.
DEC 2023	Ahmed Ghareeb, Mohammed Rashad Baker	Strategies for predictive power: Machine learning models in city-scale load forecasting	SVM, Decision Tree, Adaboost.	According to them, Bagging and Adaboost performs very well in predicting
MAY 2023	Matthew Bossart, Bri-Mathias Hodge	Machine learning for modern power distribution systems: Progress and perspectives	***Neural network , Loss function***	State that ML advanced very fast and it is applied everywhere

CHAPTER 3.

DESIGN FLOW/PROCESS

3.1Selection of Features

A dataset is a collection of data points that are related to a specific subject or area of study. It's like a giant organized table where each row represents a single observation or instance, and each column represents a specific variable or feature being measured. In order to find some pattern in the data, we need to select some specific columns from the dataset and remove the others. It can be a challenging task if our dataset has many-many columns. Here are some real-world examples of datasets:

- A dataset containing weather data for a specific location over time, with variables like temperature, humidity, and rainfall.
- A dataset recording customer purchase history at an online store, with variables like items purchased, price, and customer demographics.
- A collection of social media posts about a particular brand, with variables like text content, sentiment analysis, and user location.

To understand how households use electricity, we looked at a big collection of information, like a giant spreadsheet. This information shows how much total energy each house used over the past 3 years. Here's why this data is helpful:

Seeing the whole picture: Instead of focusing on just one appliance, we can see everything that uses electricity in the house.

Tracking changes: By looking at 3 years of data, we can see how electricity use changes throughout the year (like using more in summer for AC) and if it goes up or down over time.

Making predictions: With this data, we can use special computer programs to guess how much electricity a house might use in the future. This helps people save money and use energy wisely. By understanding how houses use electricity, we can use these computer programs to help people save money and use less energy overall!

3.2Design Constraints

For any machine learning model, there are some constraints in it. These limitations are like challenges we need to overcome to make the model work its best in the real world. Let’s take the example of our electricity management model for households.

Here are some specific challenges we might face:

- **Data Quality**: The foundation of any successful machine learning model is high-quality data. For our model, this means having reliable and detailed information on how much energy each home uses. Think of it like building a house – if the blueprints are inaccurate or incomplete, the house won't be structurally sound.

- **Simplicity:** Models must be efficient and compatible with existing household technologies. Striking a balance between predictive accuracy and computational efficiency ensures usability without requiring extensive processing power or hardware upgrades.

- **Interpretability**: Models should be interpretable, allowing users to understand the reasoning behind predictions. Transparent decision-making processes empower users to make informed choices about optimizing energy consumption.

- To mitigate distrust and enhance user engagement, our model must prioritize interpretability. Building models with transparent decision-making processes empowers users to grasp how their energy consumption influences recommendations, facilitating informed decision-making and fostering trust.

- Complexity in machine learning models often translates to resource-intensive computations, rendering them impractical for household applications. A model reliant on substantial computational power is incongruent with the everyday household infrastructure. Striking a balance between predictive accuracy and computational efficiency is crucial. Our model should seamlessly integrate with existing household technologies, like smart meters, without necessitating extravagant hardware upgrades or excessive processing power. Simplifying the model's architecture ensures real-time insights delivery, enhancing usability and accessibility for users.

Similarly, if our data is riddled with errors or missing crucial details about specific times of use or appliance usage, the model's predictions will be unreliable. Flawed data can lead the model to identify false patterns or overemphasize irrelevant factors, ultimately resulting in skewed predictions and missed opportunities for energy savings. Therefore, ensuring data quality through robust collection methods, rigorous cleaning processes, and addressing missing information is critical.

- **Privacy Concerns**: While data is essential, we can't compromise on user privacy. People naturally have concerns about sharing detailed information on their energy consumption patterns. Our model needs to be designed with privacy in mind, like using data anonymization or user consent mechanisms. This could involve looking at data at a community level or focusing on broader trends in energy use across households rather than drilling down into individual appliance usage. Striking the right balance between gathering enough information for accurate predictions and protecting user privacy is a key challenge in developing a responsible and trustworthy model.

- **Keeping it Simple**: Machine learning models can become incredibly complex, requiring massive amounts of computing power to run. While this complexity might seem desirable for achieving the most accurate predictions, in the real world of household applications, it's impractical. Imagine a model so intricate that it needs a dedicated supercomputer – it wouldn't be very useful! Our model needs to be efficient and work seamlessly with existing household technologies like smart meters. This means finding a sweet spot between model accuracy and computational efficiency, ensuring the model can deliver real-time insights without requiring extensive processing power or expensive hardware upgrades for users.

- **Understanding its Decisions**: One of the biggest hurdles in building user trust with machine learning models is their inherent "black box" nature. It can be difficult to understand why a model makes a specific prediction. In the context of our electricity management model, this lack of transparency can be problematic. Imagine a homeowner receiving a suggestion to reduce energy use, but without any explanation for why. This lack of insight can breed distrust and hinder user engagement with the model. To overcome this challenge, we need to develop models that are interpretable, meaning we can explain the reasoning behind their predictions. This allows users to understand how their energy usage patterns are influencing the model's recommendations and empowers them to make informed choices about optimizing their energy consumption.

By acknowledging and addressing these limitations, we can design a machine learning model that is not only accurate but also practical, user-friendly, and respectful of privacy.

This will pave the way for a future where machine learning empowers households to actively manage their energy use, contributing to a more sustainable future for all.

3.3 Analysis and Feature Finalization

We began by thoroughly examining the data we collected on household electricity usage. This involved looking for patterns, trends, and relationships between various factors that might influence how much energy a household uses. In our research on machine learning for household electricity management, we went through a process of the dataset to find the best candidate.

Based on our analysis, we identified specific pieces of information (features) that could be beneficial for the model. However, we had to consider several constraints before finalizing our features:

- **Data Availability**: An ideal feature for our model would be readily available for a large number of households. While some information, like weather data, can be easily obtained from weather stations or public datasets, this isn't always the case. For example, acquiring detailed information on the specific types and energy consumption patterns of appliances within each household could be challenging or even impossible to obtain on a large scale.

- **Privacy Concerns**: The collection and use of personal data necessitates a responsible approach. Features that delve deeply into a household's daily routines or activities, such as appliance usage by time of day, could raise privacy concerns. Our model's design strived to strike a balance, utilizing features that provide valuable insights without compromising user privacy.

- **Computational Power**: While including a vast amount of data might seem advantageous, features requiring extensive processing power could hinder the model's performance. Imagine a model so complex that it takes a significant amount of time to analyze data and generate predictions. In the context of real-time applications for household energy management, this would be impractical. Therefore, selecting features that can be efficiently processed by the model ensures its responsiveness and suitability for real-world use

- **Model Interpretability**: We wanted the model's reasoning to be understandable. Features that are too complex might make it difficult to explain the model's predictions to users.

By analyzing our data and considering these constraints, we determined that **"total energy consumption"** emerged as a strong and versatile feature for our model. This readily available metric encompasses a household's overall electricity use, providing valuable insights without the need for intrusive data collection or complex computations. Furthermore, focusing on total energy consumption allows the model to remain interpretable, enabling users to

understand how their overall energy use patterns influence the model's predictions. In conclusion, prioritizing features that met these criteria facilitated the development of an efficient, user-friendly model capable of providing valuable insights for household energy management, paving the way for a more sustainable future.

3.4 Design Flow

Our project on using machine learning for household electricity management follows a well-defined design flow:

- **Data Collection and Preprocessing**: The first step involves gathering data on household electricity usage. This could come from smart meters, utility bills, or surveys. Once collected, the data needs cleaning and organization. This might involve removing errors, handling missing values, and ensuring consistency in formatting.

- **Feature Engineering and Selection**: Here, we delve into the data to identify the most impactful features that influence electricity consumption. This involves analyzing historical trends, seasonal variations, and potential correlations with external factors like weather. Through a process called "feature engineering," we might create new features based on existing data (e.g., average daily consumption). Finally, considering constraints like data availability, privacy, and model interpretability, we select the most relevant features for our model.

- **Model Training and Evaluation**: With the features chosen, we build the machine learning model. This involves selecting a suitable machine learning algorithm and training it on the prepared data. As the model learns, we evaluate its performance on unseen data to assess its accuracy and generalizability. This iterative process might involve refining the model or feature selection until we achieve a satisfactory level of performance.

- **Deployment and Monitoring**: Once the model is trained and validated, it's ready for deployment. This could involve integrating it with existing smart home systems or creating a user-friendly interface for households to interact with the model. Finally, we monitor the model's performance in real-world use and make adjustments as needed to ensure it continues to provide accurate and valuable insights for optimizing electricity consumption.

3.5 Design selection

For our project on household electricity management using machine learning, we opted for a unique design selection that leverages the power of signal processing and curve fitting techniques. This approach deviates from the traditional supervised learning methods often employed in such projects, offering a distinct set of advantages tailored to our specific goals.

- **3.5.1 Signal Processing**

Signal Processing is one of the most powerful fields in mathematics. It has been in use since many years. Radios and TVs all rely on the methods of signal processing to give the results.

It's used in the following but not limited to them:

- **Identifying Usage Patterns:** By analyzing the "total energy consumption" data over time (daily, weekly, monthly), you can utilize techniques like moving averages or seasonal decomposition to identify recurring patterns in energy use. This might reveal peak consumption periods (e.g., mornings and evenings) or seasonal variations due to weather or holiday activities.

- **Exploring Consumption Fluctuations:** Signal processing techniques like spectral analysis can be applied to "total energy consumption" data to understand how energy use fluctuates across different time scales. This might reveal dominant frequencies related to specific appliance usage patterns (e.g., high-frequency spikes for short-burst appliances like microwaves, lower frequencies for continuously running appliances like refrigerators).

- **Correlations and Relationships:** Techniques like cross-correlation can be used to explore the relationship between "total energy consumption" and "energy used." This might reveal how individual usage events (reflected in "energy used") contribute to the overall consumption picture ("total energy consumption").

- **3.5.2 Curve Fitting for Prediction**

Once the signal processing techniques have unveiled the hidden patterns and features within the consumption data, we can leverage the power of curve fitting to predict future electricity usage. Curve fitting is a mathematical technique that aims to find a function that best represents a set of data points. In our case, we can utilize curve fitting algorithms to identify the mathematical relationship between the extracted features (like frequency components or daily usage patterns) and historical consumption data.

We employed two methods for curve fitting:

- **Polynomial Curve:** Polynomial fitting involves fitting a mathematical equation, specifically a polynomial, to your data points. Imagine plotting your "total energy consumption" data over time. A polynomial fitting technique can identify a curve that best represents the overall trend in the data. This can be particularly valuable for uncovering seasonal variations in energy use. For example, a polynomial might reveal

a gradual increase in consumption during summer months due to air conditioning usage, followed by a decrease in winter.

- **Sinusoid Curve:** Sinusoidal fitting involves fitting a sine wave to your data. This approach is particularly well-suited for identifying cyclic patterns within your features. Remember how we discussed "energy used" potentially reflecting individual usage events? By fitting a sine wave to this data, you might uncover daily, or weekly patterns related to specific appliance usage or routine activities. For instance, you might see a recurring sinusoidal pattern in "energy used" data, reflecting peak usage periods like mornings and evenings when appliances like coffee makers or toasters are used.

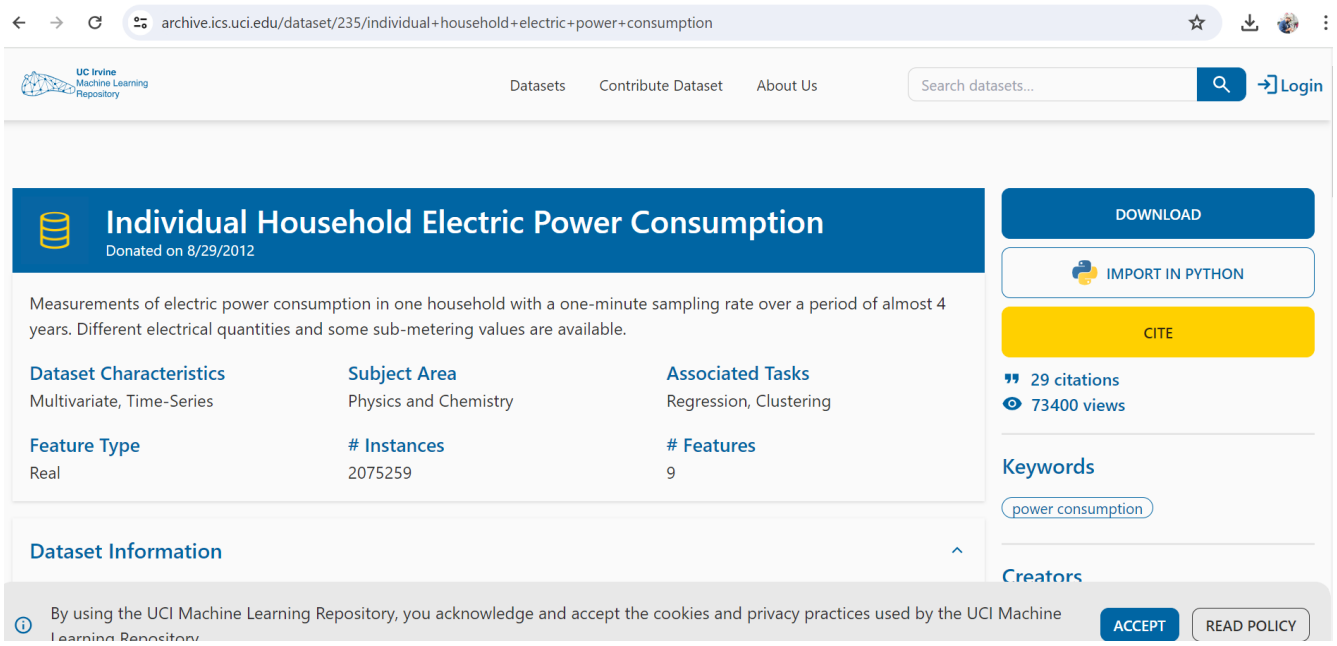
This allows us to create a mathematical model that captures the essential trends and variations in electricity usage. With this model in hand, we can then make predictions about future consumption patterns based on new input data. Imagine having a formula that can accurately predict how much energy a household will likely use tomorrow based on its historical consumption patterns and the identified trends revealed by signal processing. This predictive capability becomes the cornerstone for providing actionable insights for optimizing electricity consumption in households.

3.6

CODE

Demand Side Management for Electricity Consumptio

- The project does make use of dataset from https://doi.org/10.24432/C58K54 and is freely available on the address https://archive.ics.uci.edu/dataset/235/individual+household+electric+power+consumption
- This dataset is a time series dataset.
- The size of dataset is around 120MB!



Here's a paraphrased version:

Sqlite3: A Python library integrated for managing SQLite databases, offering a straightforward approach to interact with them, particularly suited for smaller-scale projects and initial development stages.

Plotly: Renowned for crafting interactive and professional-grade plots, this library encompasses a wide array of chart types, extending its capabilities to building dynamic dashboards and web-based visualizations.

Numpy: Serving as the cornerstone for scientific computation in Python, Numpy empowers users with extensive support for large, multi-dimensional arrays and matrices, complemented by a rich suite of mathematical functions to streamline array operations.

Scipy: Enhancing Numpy's functionality, Scipy serves as a comprehensive library for scientific and technical computing, furnishing additional modules for optimization, integration, interpolation, linear algebra, signal processing, and beyond.

Pandas: Revered for its prowess in data manipulation and analysis, Pandas introduces intuitive data structures like Series and DataFrame, tailored for seamless handling of structured data. Widely adopted for tasks spanning data preprocessing, cleansing, transformation, and in-depth analysis.

Matplotlib: A versatile library cherished for its expansive plotting capabilities, Matplotlib is indispensable for crafting static, animated, and interactive visualizations. Its vast repertoire of plotting functions and customization options renders it indispensable for generating high-quality figures across scientific research, data exploration, and presentation endeavors.

These libraries, when used collectively, empower Python developers to tackle a diverse range of tasks, from data analysis and visualization to database management, catering to varied needs across different domains.

Necessary Imports

```
import sqlite3
import matplotlib.pyplot as plt
import numpy as np
import plotly.express as exp
import pandas as pd
from vedo import pyplot

con = sqlite3.connect("./Dataset/Database")
```

Get some data from the dataset to get some initial insights.

```
cur = con.cursor()

avg_consumption = cur.execute("select time, avg(Global_intensity) from household_power_consumption GROUP by time;")
avg_consumption = np.array( avg_consumption.fetchall() )
total_points = len(avg_consumption)

df = pd.DataFrame({"time": avg_consumption[:, 0], "Avg. Consumption (in KJ)": np.array(avg_consumption[:, 1], dtype=np.f
```

This code initiates a connection to an SQLite database and runs a SQL query to compute the average consumption grouped by time from a table labeled "household_power_consumption". It then retrieves the outcomes, transforms them into a NumPy array, calculates the total number of data points, and constructs a DataFrame to hold the results. This DataFrame consists of two columns: "time" and "Avg. Consumption (in KJ)"

```
exp.line(df, x="time", y="Avg. Consumption (in KJ)" , width=800, height=600)
# df['Avg. Load'].dtype
```

Checking monthly energy consumption

Average energy consumption by hour

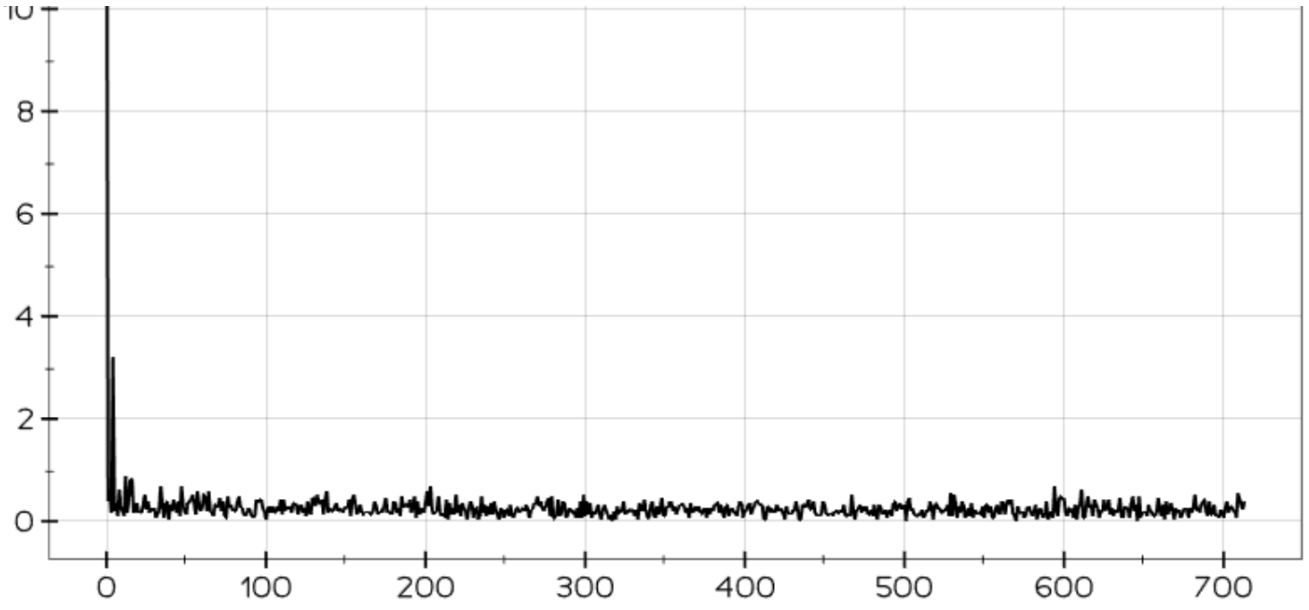
Fourier Analysis

```
from scipy.fft import fft, fftfreq
from scipy.signal import convolve
cur = con.cursor()
data = cur.execute('SELECT Global_intensity from household_power_consumption where time = "20:00:00" and Global_intensit
data = cur.fetchall()

# Converting it to Numpy array
```

```
# Converting it to Numpy array
data = np.array(data,)
data_length = data.shape[0]
data = data.reshape(data_length,)

# Just for single use only
df = pd.DataFrame({"Recording Number": np.arange(data_length), "Load": data})

exp.line(df, x="Recording Number", y="Load", width=800)
```

The provided code connects to an SQLite database and retrieves the "Global_intensity" data for a specific time from the "household_power_consumption" table. After fetching the results, it converts them into a NumPy array and reshapes them to a one-dimensional format. Then, it creates a DataFrame named "df" for visualization, containing columns for "Recording Number" and "Load". Finally, the data is plotted.

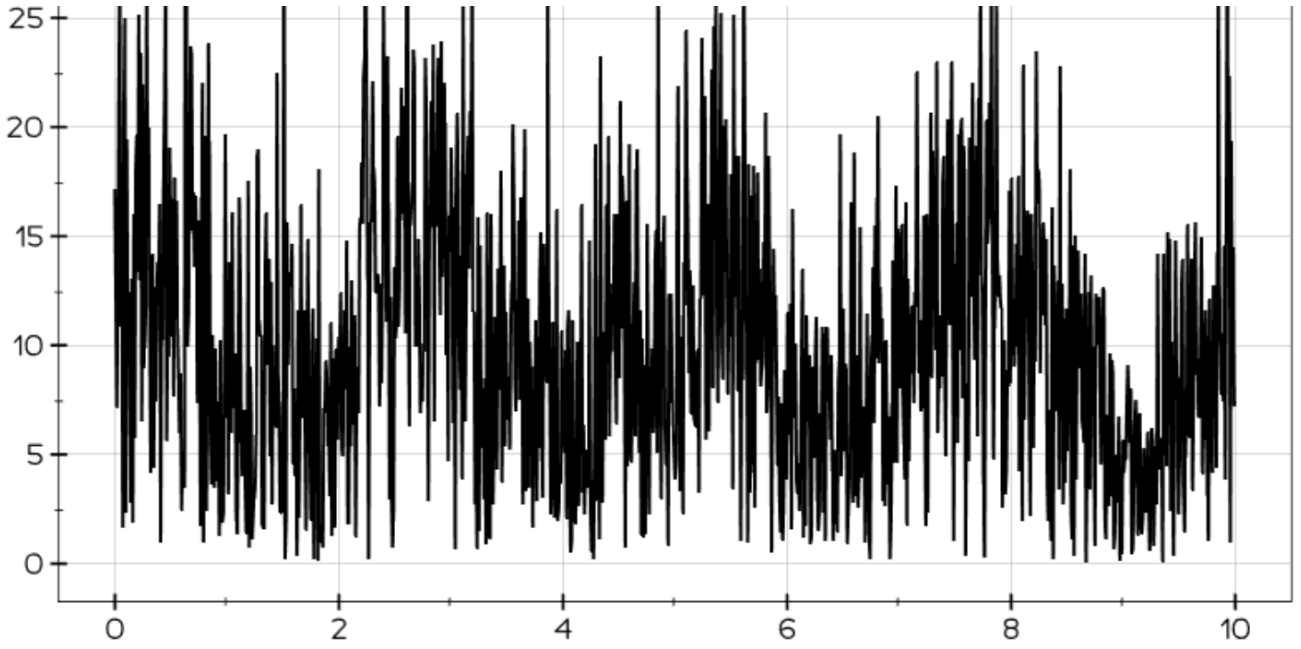
```
## The fft algorithm is applied here.

fft_data = fft(data)
frequencies = data_length * fftfreq(data_length)

p = pyplot.plot(np.abs(frequencies), 2 * np.abs(fft_data) / data_length )
p.show()
```



Frequency and Amplitudes we get from the signal

```
# Now we can find the top frequencies and amplitude making the given signal.

fft_data_mean = np.mean(np.abs(fft_data) )
fft_std = np.std(np.abs(fft_data) )

fft_data_cleaned = fft_data[np.abs(fft_data) > 1 / 2 * fft_data_mean]      # Trying to clean the signal. This signa


""" Now recreating the signal from the cleaned signal """
clean_signal = np.fft.ifft(fft_data_cleaned)
p = pyplot.plot( np.linspace(0, 10, len(clean_signal) ), np.abs(clean_signal), ylim = [0, 35] )
p.show()
```

The code calculates the mean and standard deviation of the FFT data, then filters out values below a specified threshold to clean the signal. Next, it reconstructs the signal from the cleaned FFT data using inverse FFT. Finally, it visualizes the cleaned signal

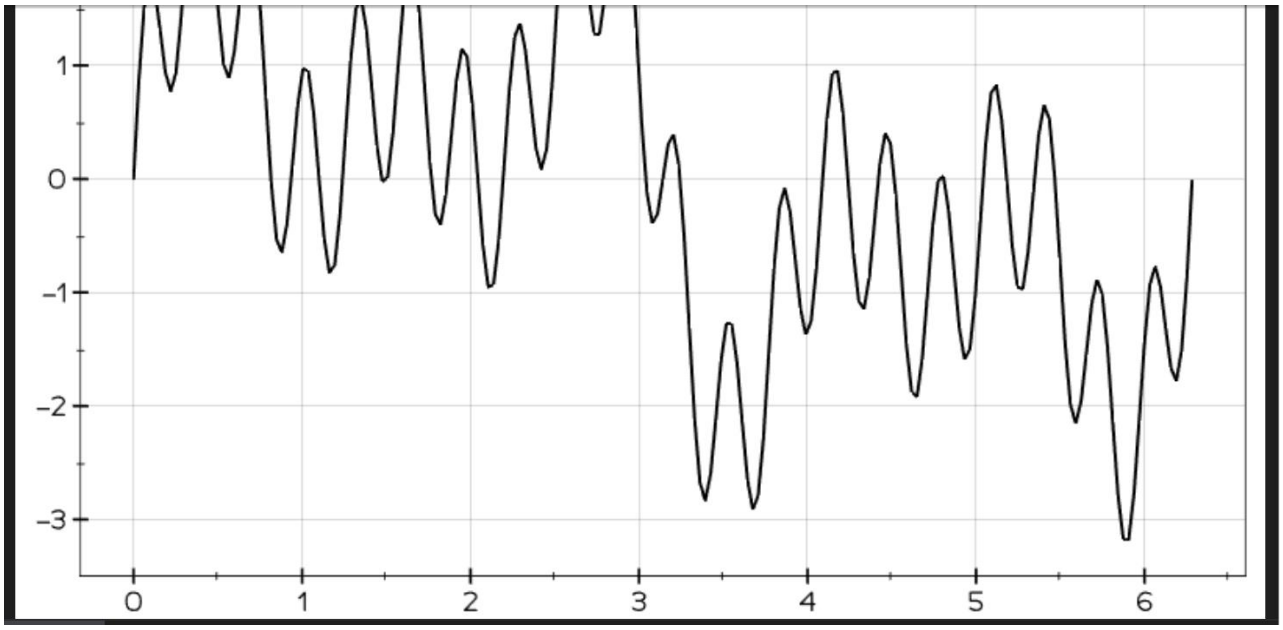


Plot created by fft

Basic FFT Demo

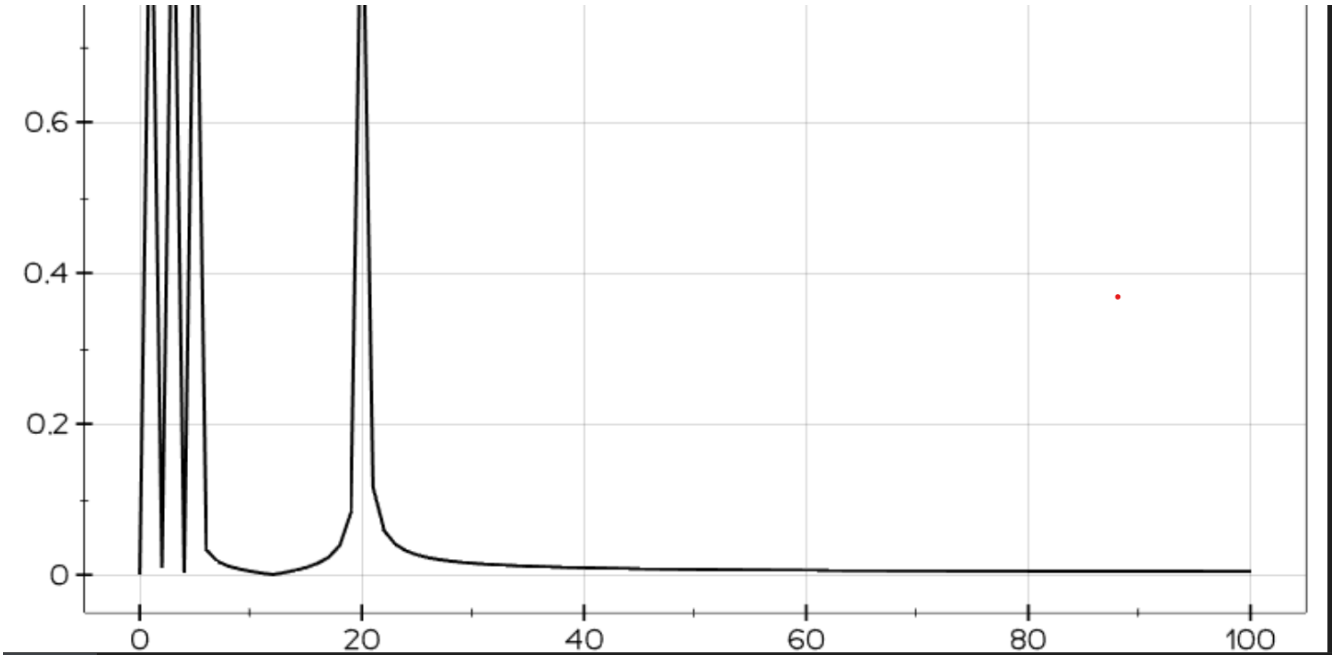
```
import numpy as np
from scipy.fft import fft, fftfreq
from vedo.pyplot import plot

samples = 201
x = np.linspace(0, 2 * np.pi, samples)
y = np.sin(3*x) + np.sin(5*x) + np.sin(x) + np.sin(20*x)     # The input signal

p = plot(x, y )
p.show()
```

The provided code imports required libraries, sets the number of samples, generates a complex input signal, and visualizes it using the vedo library's plotting function called PLOT.



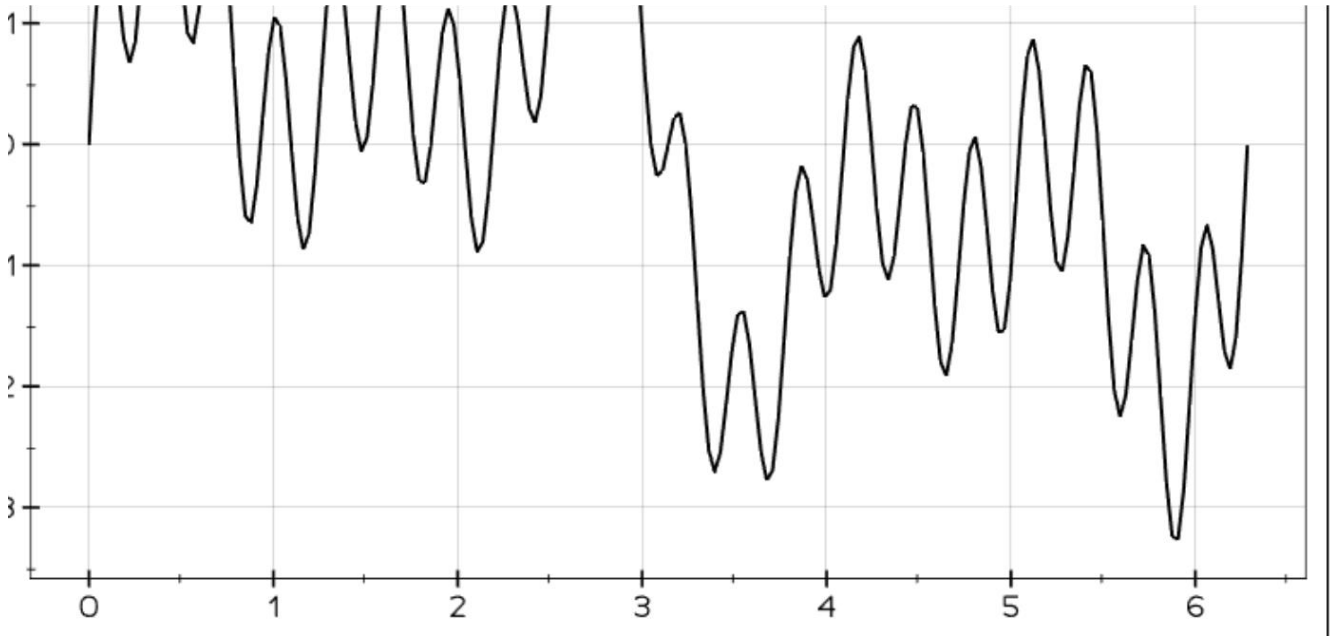
Original Plot

```
    # Since FFT frequencies are repeated, therefore we can discard other half values
    fft_y = fft(y)   # FFT applied
    freq = samples * fftfreq(samples)

    p = plot(np.abs(freq), 2 * np.abs(fft_y) / samples)
    p.show()
```



Plot of Amplitude vs Frequency

```
sorted_frequencies = np.argsort( 2 * np.abs(fft_y[:samples // 2]) / samples )[::-1][:5]
sorted_amplitudes = (2 * np.abs(fft_y) / samples)[sorted_frequencies]


# Plotting the function
samples = 201
x = np.linspace(0,2 * np.pi, samples)
y = np.zeros(samples)
for i in range(len(sorted_frequencies) ):
    y += sorted_amplitudes[i] * np.sin(sorted_frequencies[i] * x)

p = plot(x, y)
p.show()
```

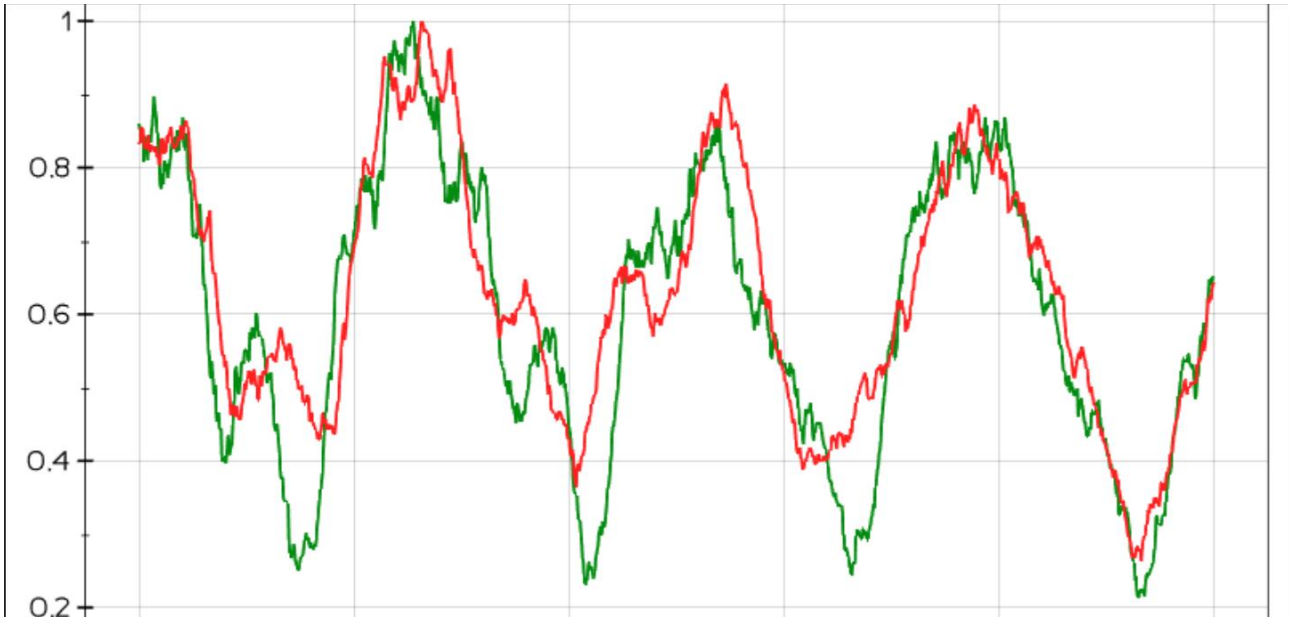


Recreated Plot

```
    # Trying the correlation on the data.
    # Also convolution on the signal to better gain an understanding on the signal.

    from scipy.signal import convolve
    wave1 = data / np.max(data)
    wave2 = np.abs(clean_signal) / np.max(np.abs(clean_signal) )

    wave3 = np.ones(50)
    output = convolve(wave1, wave3, mode = "valid")
    output2 = convolve(wave2, wave3, mode = "valid")


    # # This is the signal plot showing the closeness.
    # p = pyplot.plot(np.linspace(0, 10, len(wave1 )), wave1, c="g", ylim = [0, 1] )
    # p += pyplot.plot(np.linspace(0, 10, len(wave2 )), wave2, c = 'r', ylim = [0, 1])
```

```
# # This is the signal plot showing the closeness.
# p = pyplot.plot(np.linspace(0, 10, len(wave1 )), wave1, c="g", ylim = [0, 1] )
# p += pyplot.plot(np.linspace(0, 10, len(wave2 )), wave2, c = 'r', ylim = [0, 1])


# This is the correlation plot.
p = pyplot.plot(np.linspace(0, 10, len(output )), output / np.max(output), c = "g", ylim = [0, 1] )      # Original s
p += pyplot.plot(np.linspace(0, 10, len(output2 )), output2 / np.max(output2), c = "r", ylim = [0, 1])      # Clean s

p.show()
```

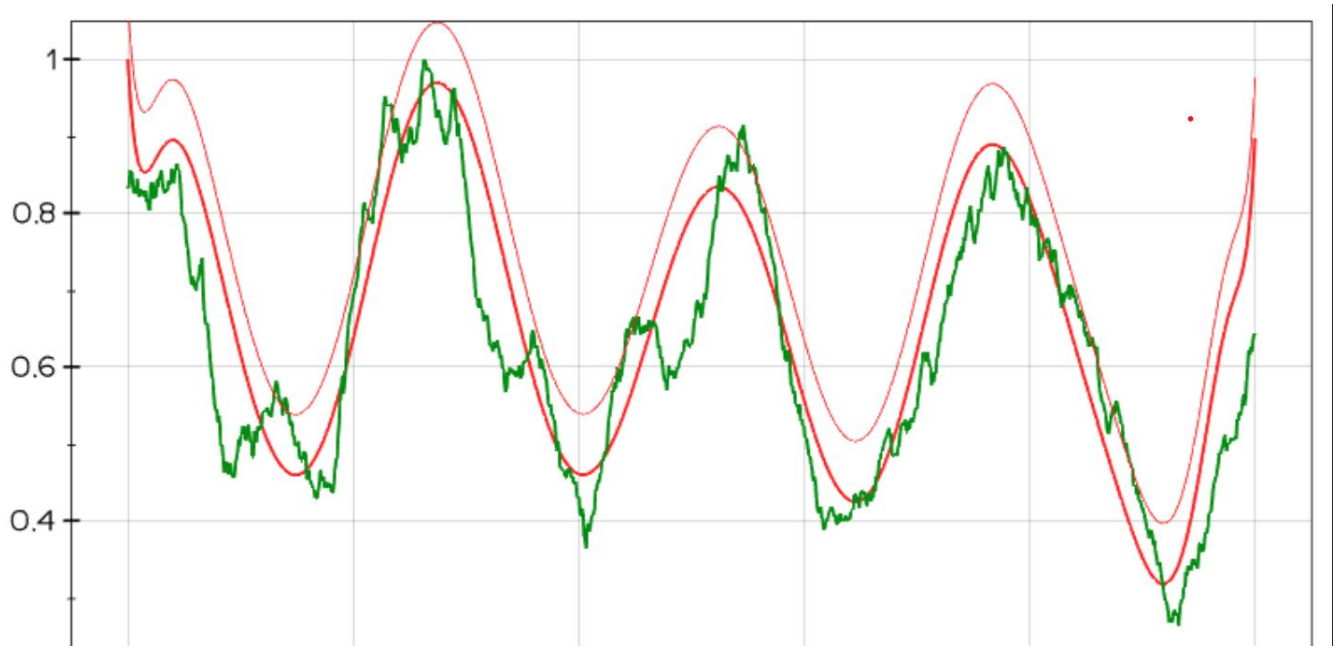
The provided code normalizes two sets of data, conducts a convolution operation between them using a specified kernel, and then plots the correlation between the resulting signals. This correlation plot illustrates the relationship between the two original signals after convolution.



Fitting polynomial to data for approximation

```
# This fit method used Numpy Library for approximation.

a = np.polyfit(np.linspace(0, 10, len(wave2 )), wave2 , 20)      # Returns a * x**20 + b * x**19 + c * x**18 ...

x = np.linspace(0, 10, len(output2) )
y = np.zeros(len(x) )
for i in range(len(a) ):
    y += a[i] * x ** (len(a) - 1 - i)

# Now normalize y
y = y / np.max(y)

# Plot the fit with original data
p = pyplot.plot(x, y, c = 'r', ylim = [0, 1])
p += pyplot.plot(x, output2 / np.max(output2), c = 'g', ylim = [0, 1] )
```

```
# We can also plot the deviation along with the fit polynomial.
deviation = np.std(y - output2 / np.max(output2) )

p += pyplot.plot(x, y + deviation, c = 'r', ylim = [0, 1], lw = 1, xtitle = "Power fluctuation")
# p += pyplot.plot(x, y - deviation, c = 'r', ylim = [0, 1], lw = 0, marker = 'o' )
p.show()
```



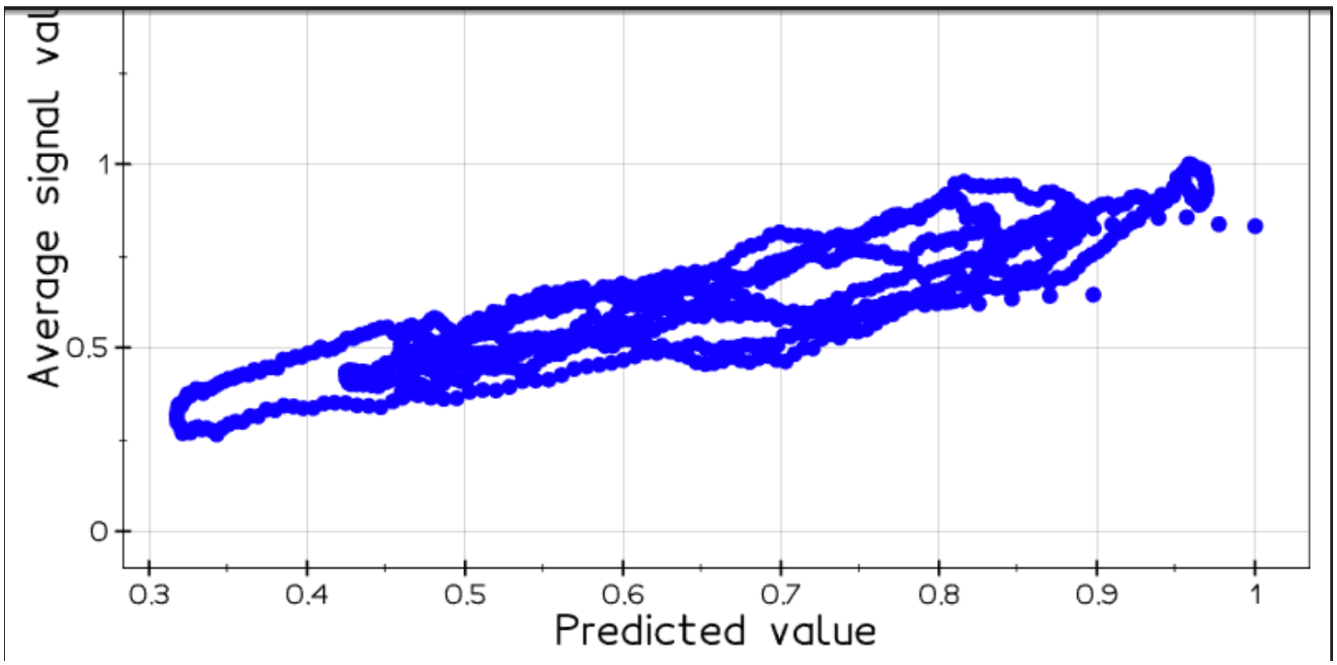
With 14 percent error, we can predict the average consumption at the given time.

```
deviation = np.std( (y - output2) / np.max(output2) )
print("The standard deviation is ->", deviation * 100, "percent" )
# print(len(y), len(output2) )
```

Python

```
The standard deviation is -> 14.516234179029322 percent
```

```
# Since the graph showed some correlation between data of mean + standard deviation and cleaned signal

p = pyplot.plot(y , output2 / np.max(output2) , c='b', lw = 0, xtitle = "Predicted value", ytitle = "Average signal val
p.show()
```



Relation between predicted and average rolling mean value

Fitting a line between predicted value and average rolling mean value

```
# Finding the best fit line for our data

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

x_train, x_test, y_train, y_test = train_test_split(y + deviation, output2)   # Split the data into test train split

model = LinearRegression()
model.fit(x_train.reshape(-1, 1), y_train.reshape(-1, 1) )
```

```
    model = LinearRegression()
    model.fit(x_train.reshape(-1, 1), y_train.reshape(-1, 1) )

    score = model.score(x_test.reshape(-1, 1), y_test.reshape(-1, 1) )
    print( "The score of our model in TEST DATA is :", score)
```

25]

```
The score of our model in TEST DATA is : 0.6443484041472181
```

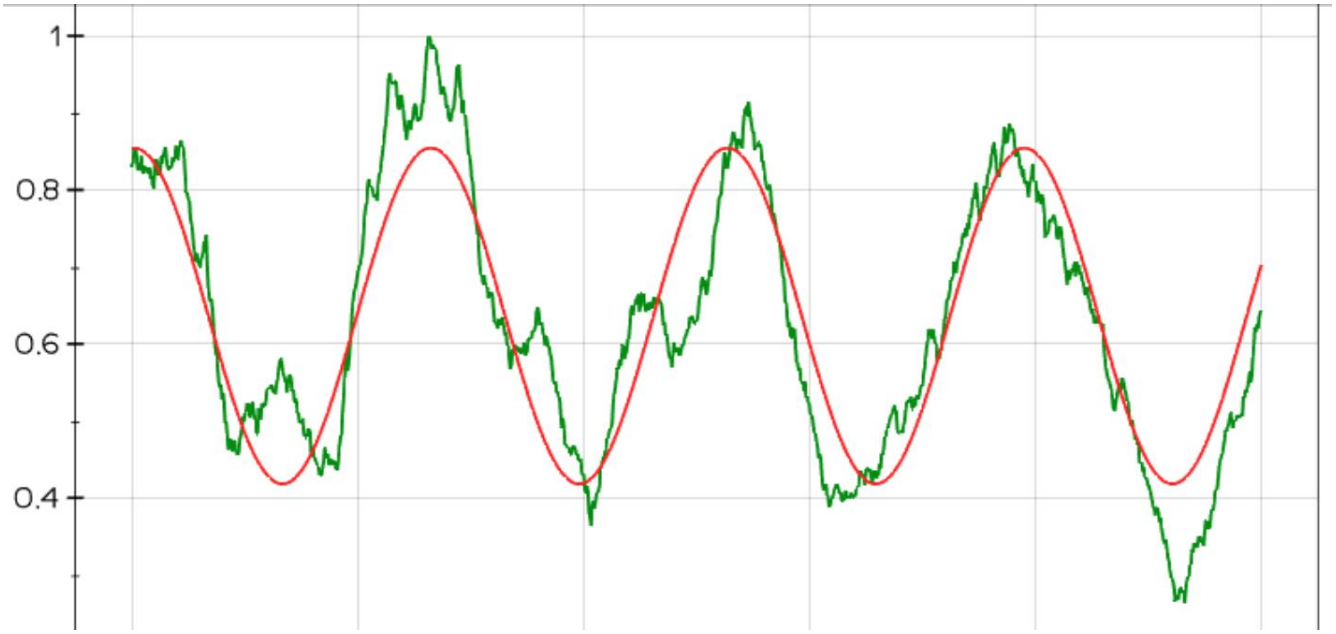
```
    score = model.score(x_train.reshape(-1, 1), y_train.reshape(-1, 1) )
    print( "The score of our model in TRAINING DATA itself is :", score)
```

26]

```
The score of our model in TRAINING DATA itself is : 0.6647188705198754
```

Since our model is a sinusoidal curve. We can also try to fit a sinusoid curve using scipy library.

```
from scipy.optimize import curve_fit

def func(x, a, b, c, d):
    return a * np.sin(b * 2 * x + d) + c

popt, pcov = curve_fit(func, x, output2 / np.max(output2) )


optimized_curve = func(x, *popt)

p = pyplot.plot(x, output2 / np.max(output2), c = 'g', ylim = [0, 1] )
p += pyplot.plot(x, optimized_curve, c = "r", ylim = [0,1] )
p.show()
```




```
    result = np.std( (optimized_curve - output2) / np.max(output2) )
    print("With this method, the standard deviation is", result * 100, "percent" )
```

```
With this method, the standard deviation is 12.23422340407231 percent
```

In future,We can apply neural networks to better fit the curve.....

We can also try RNN and LSTM for this time series data.

```
[30]  a = np.random.randint(0, 1000,(5))
```

Python

```
[31]  a
```

Python

```
array([770, 255, 151, 585, 736])
```

```
    for i in a:
        print("rolling mean ->", output2[i] / np.max(output2) )
        print("optimized curve ->", y[i])
        print("sinusoid curve is ->", optimized_curve[i])
```

```
rolling mean -> 0.6490467980139836
optimized curve -> 0.6491874990598964
sinusoid curve is -> 0.663702726764576
rolling mean -> 0.9580142120329352
optimized curve -> 0.771266565374536
sinusoid curve is -> 0.9337975302260281
rolling mean -> 0.526605342121285
optimized curve -> 0.42234807791521317
sinusoid curve is -> 0.594546826466158
rolling mean -> 0.8412426205655331
optimized curve -> 0.7745302354453018
sinusoid curve is -> 0.9233412549200534
rolling mean -> 0.5489504851378759
optimized curve -> 0.4906827830462119
```

Certainly! These numbers represent scores indicating the performance of different types of curves:

1. Rolling Mean Curve:
   - Achieves a score of around 0.65.
   - This curve likely reflects a smoothed or averaged version of data, with a moderate performance level indicated by the score.

2. Optimized Curve:
   - Also scores approximately 0.65.
   - This curve represents a fine-tuned or optimized model, showing similar performance to the rolling mean curve.

3. Sinusoid Curve:
   - Scores slightly higher, at about 0.66.
   - This curve reflects a periodic function, such as a sine wave, and shows slightly better performance compared to the other curves.

These scores are indicative of how well each curve fits the underlying data, with higher scores suggesting better alignment. However, the specific evaluation metric or context in which these curves are used would provide a clearer understanding of their performance.

CHAPTER 4.

METHODOLOGY

- **This architecture involves several steps:**
- **Data Collection:**
- **Data Preprocessing:**
- **Feature Engineering:**
- **Machine Learning Model Training:**
- **Model Evaluation:**
- **Deployment:**
- **User Interface:**
- **Monitoring and Maintenance:**

A. Downloading the dataset Before we can do any inference on average electricity consumption, we need some data which can help us do statistics. The dataset should be large enough that we can see the yearly variation in the electricity consumption of the household. Since this kind of dataset will take a large space the disk and will be computationally expensive, we used a powerful enough machine to process the data we were exploring. The dataset used in this paper is "Individual household electricity consumption". This dataset contains time series of 2,075,259 points. The data collected is between December 2006 and November 2010.

B. Feature Selection This step is a crucial step in our method, or any other machine learning project. In this step, we identify the potential features we need or can use in our method. The machine learning methods rely heavily on the quality of features in our data. For a data processing task, feature is the column of the data which we can make use of, for our training. This step requires us to find relevant features in the data which are relevant to our problem. This often requires expertise in the domain and good understanding of our data. The features used in our problems are the time and global intensity. These both are continuous time-series data and are a reliable source for predicting consumption at the moment. Feature selection comes under the domain of feature engineering. Feature engineering plays and important role in the implementation of any machine learning project. By carefully selecting and transforming the data, we can increase the effectiveness of our model.

C. Preprocessing the data Any machine learning method relies heavily on the quality of the data. It can be further simplified: Machine learning models require clean and well-structured data for

good output. Only loading data and doing analytics just doesn't work. For example, NumPy requires the datatypes to be "int", or a "float" to calculate the mean and other quantitative overview of the data. By "int" and "float", we mean that the datatype can be any "int" or "float", it can also be "int8", "int16", "float16", "float32" or any other datatype other than "None" or "str". But, if the data contains even a single "None" type, it will change the datatype of the whole data to be "str". And, in "str" datatype, we cannot do general analytics like mean, deviation and variance. The size of our dataset, when uncompressed, as a plain Comma Separated Values (CSV) file was around 130MB which is huge compared to most of the other models. For performance reasons, we made the use of SQLite database to decrease the time taken in calculation in the data. It also has an in-built support in the Python language so we used it for further calculations.

D. Data Exploration Data Exploration is a method of finding patterns in the data by simply visualizing the data using a software. This step is sometimes also known as Visual Data Mining because it is easy to find the patterns in data by making a graph in the data. There are a plenty of data visualization libraries present in the Python ecosystem but we chose the Vedo library to be the best candidate. Vedo is a wrapper library around another package called VTK. VTK stands for Visualization Toolkit and used for processing the data in bulk. It contains many types of plots and it can also represent 3D very efficiently

This section outlines the methodological approach employed in our project for developing a machine learning model to optimize electricity management in households. Focusing on all the important steps required, we will see all the key steps required.
The steps are as follows:

4.1 Data Acquistion and Preprocessing

Variable Name	Role	Type	Description	Units	Missing Values
Date	Feature	Date			no
Time	Feature	Categorical			no
Global_active_power	Feature	Continuous			no
Global_reactive_power	Feature	Continuous			no
Voltage	Feature	Continuous			no
Global_intensity	Feature	Continuous			no
Sub_metering_1	Feature	Continuous			no
Sub_metering_2	Feature	Continuous			no
Sub_metering_3	Feature	Continuous			no

Table Columns and other information

This is a multistep process where we download or create a dataset from a reliable source. This step contains the following:

4.1.1 Data Source: In this step, we gather data on household electricity consumption. This may involve partnering with utility companies or utilizing smart meter data from participating households. But for our case we downloaded the data from https://archive.ics.uci.edu/dataset/235/individual+household+electric+power+consumption . This dataset comes in a compressed ZIP format. ZIP is one of the most popular compression formats and used widely in the industry. It is an open-source project supported by MNC's and maintained by individuals.

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	16/12/2006	17:24:00	4.216	0.418	234.84	18.4	0.0	1.0	17.0
2	16/12/2006	17:25:00	5.36	0.436	233.63	23.0	0.0	1.0	16.0
3	16/12/2006	17:26:00	5.374	0.498	233.29	23.0	0.0	2.0	17.0
4	16/12/2006	17:27:00	5.388	0.502	233.74	23.0	0.0	1.0	17.0
5	16/12/2006	17:28:00	3.666	0.528	235.68	15.8	0.0	1.0	17.0
6	16/12/2006	17:29:00	3.52	0.522	235.02	15.0	0.0	2.0	17.0
7	16/12/2006	17:30:00	3.702	0.52	235.09	15.8	0.0	1.0	17.0
8	16/12/2006	17:31:00	3.7	0.52	235.22	15.8	0.0	1.0	17.0
9	16/12/2006	17:32:00	3.668	0.51	233.99	15.8	0.0	1.0	17.0
10	16/12/2006	17:33:00	3.662	0.51	233.86	15.8	0.0	2.0	16.0
11	16/12/2006	17:34:00	4.448	0.498	232.86	19.6	0.0	1.0	17.0

Example Data

1.1. **Data Cleaning:** The dataset is humongous in size. If the dataset is large in size, it might be the case that dataset contains null values in it. Data cleaning is a crucial initial step in our project. Imagine building a house on a foundation riddled with cracks. Similarly, inaccurate or inconsistent data can lead to unreliable predictions from our machine learning model. The data cleaning process meticulously addresses missing values, outliers, and inconsistencies in timestamps or data formats. This ensures the data accurately reflects real-world electricity consumption patterns, laying a solid foundation for the model to learn from and generate trustworthy predictions for optimizing household energy management. The screenshot below shows the example of missing features in the dataset.

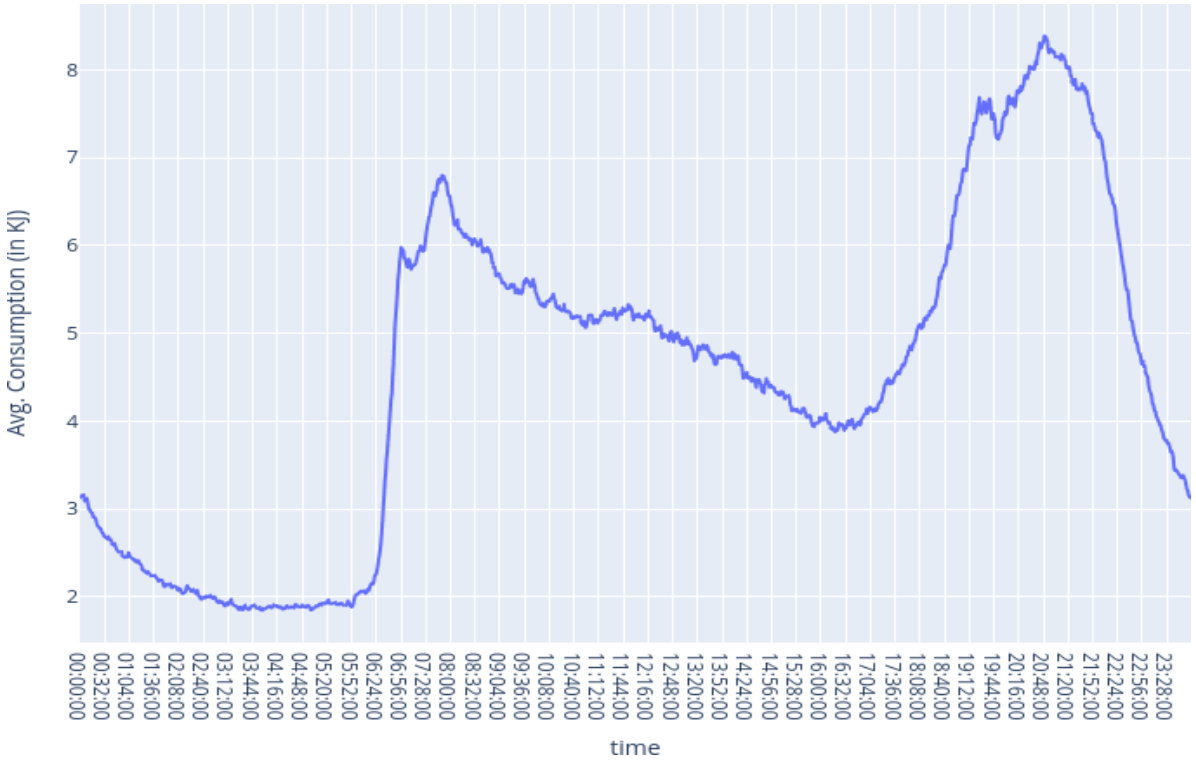
```
1  select Date from household_power_consumption WHERE Global_reactive_power is "?";
```

	Date
1	21/12/2006
2	21/12/2006
3	30/12/2006
4	30/12/2006
5	14/1/2007

```
Execution finished without errors.
Result: 25979 rows returned in 98ms
At line 1:
select Date from household_power_consumption WHERE Global_reactive_power is "?";
```

Example of missing features

1.2. **Feature Engineering**: Feature engineering is the art of transforming raw data into a format that a machine learning model can effectively understand and utilize for prediction. In our project, this involves strategically transforming and manipulating the existing features – "total energy consumption" and "energy used" – to create even more informative representations. An example of the feature is the average electricity usage by the house. These features also need to be extracted from the dataset itself. In our dataset we have 9 features of interest, out of these features we need to extract the features that can help us in finding some interesting feature for further tasks.

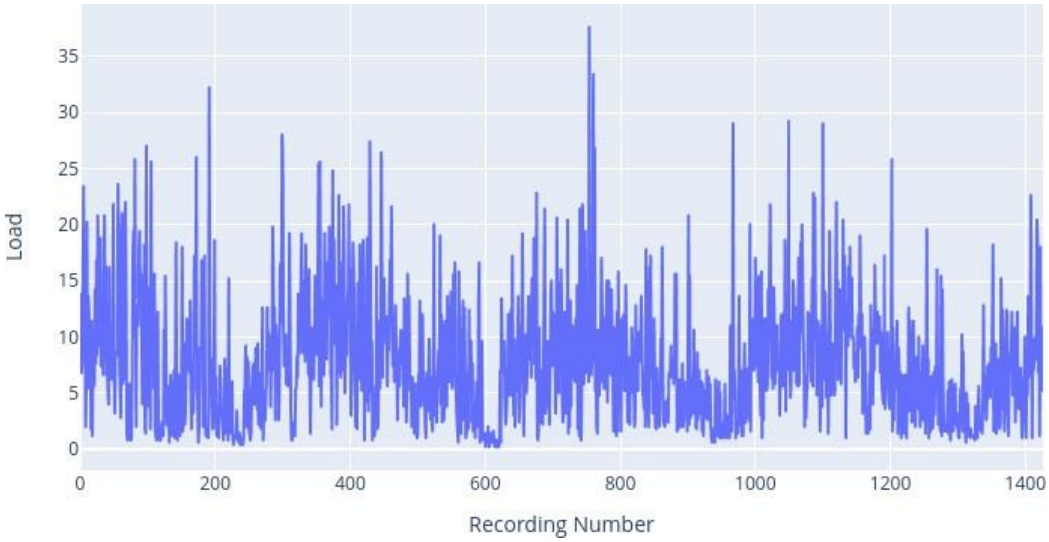


Average consumption of power with respect to time

4.2 Signal Analysis

Signal Processing is a fascinating field that deals with the analog signals and digital signals. A signal is a time varying quantity that carries information. This can be something as simple as a light being turned on or off, or it can be complex data like the voltage fluctuations that represent sound in a speaker. The key is that the signal changes over time, and these changes encode the information we want to transmit. In electronics, these signals are often electrical quantities like voltage or current, but they can also be sound waves, light waves, or any other physical phenomenon that can be varied to

represent information. DSP is all about extracting information, optimizing transmission, and enhancing the quality of signals. It's used in countless applications, from noise reduction in headphones to compressing videos for faster loading on the internet. Even medical imaging relies on signal processing to create detailed pictures of our insides for doctors. The next time you use a smartphone, listen to music, or watch a video, remember the invisible power of signal processing working behind the scenes.



The original signal

Some techniques of signal processing are as follows:

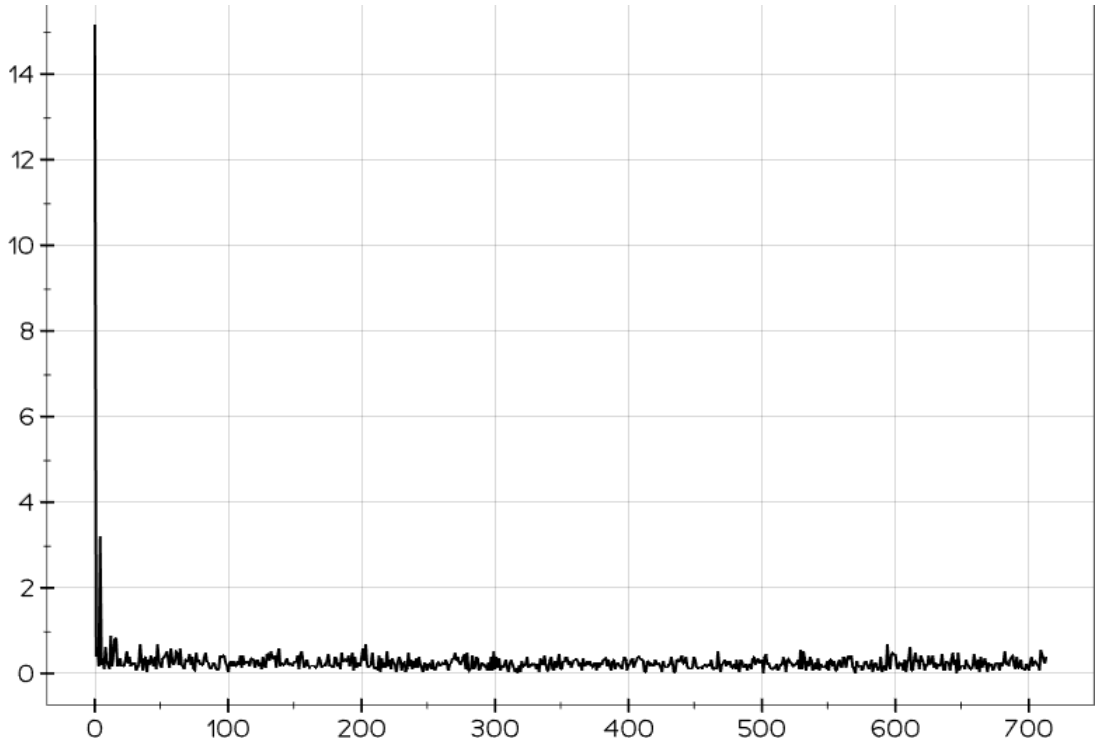
- **Spectrometry:** This technique breaks down a complex signal into its constituent frequencies, much like separating light into a rainbow. Techniques like Mass Spectrometry and Fourier Transform Infrared Spectroscopy (FTIR) achieve this by using different methods to interact with the signal and reveal its hidden frequency components. This information is crucial for tasks like material identification, disease diagnosis, and analyzing the chemical composition of objects.
- **Fast Fourier Transform (FFT):** As mentioned earlier, the FFT is a workhorse in signal processing. It's a highly efficient algorithm for performing spectrometry, calculating the frequency content of a signal with remarkable speed. This allows us to analyze large datasets in real-time, making it essential for applications like audio and image compression, noise reduction in headphones, and analyzing brain waves in EEGs.
- **High-Pass Filter (HPF) and Low-Pass Filter (LPF):** These filters act as frequency gatekeepers, allowing only specific frequency ranges to pass through while attenuating others. HPFs permit only high-frequency components to pass, effectively removing low-frequency noise or unwanted bass in audio signals. Conversely, LPFs let only low frequencies through, useful for removing high-frequency noise or sharpening blurry images by eliminating high-frequency details.

- **Sampling and Quantization:** The real world operates in continuous time, but computers deal with discrete values. Sampling involves converting a continuous signal into a series of discrete data points, essentially taking snapshots at specific time intervals. Quantization further reduces the complexity by assigning these data points discrete values. These techniques are crucial for storing and processing signals digitally.
- **Convolution:** This mathematical operation plays a vital role in signal processing. It essentially describes how one signal influences another. Convolution is used extensively in filtering by employing specific filter kernels to manipulate the signal's frequency content. It also finds applications in image processing tasks like blurring and edge detection.

4.3 Fast Fourier Transform

Signal processing relies heavily on a powerful tool called the Fast Fourier Transform (FFT). Imagine you have a recording of music, which can be thought of as a single complex signal over time. But music is actually made up of many different frequencies, like the base notes, the mids for vocals, and the highs for cymbals. The Fourier Transform takes this time-domain signal and decomposes it into its frequency components, revealing which frequencies are present and at what strength.

This is fantastic for many reasons. By understanding the frequencies, we can isolate instruments, remove unwanted noise at specific frequencies (like a constant hum), or even equalize the sound by adjusting specific frequency ranges. However, the standard way to do this decomposition, the Discrete Fourier Transform (DFT), can be very slow for large signals. Here's where the Fast Fourier Transform comes in. It's a clever algorithm that calculates the same result as the DFT, but in a much faster way. The FFT breaks down the complex signal into smaller, more manageable chunks and then reassembles the pieces to get the frequency information. This dramatically reduces the number of calculations needed, making it possible to analyse large audio files, images, or other data sets in real-time.



FFT of the signal

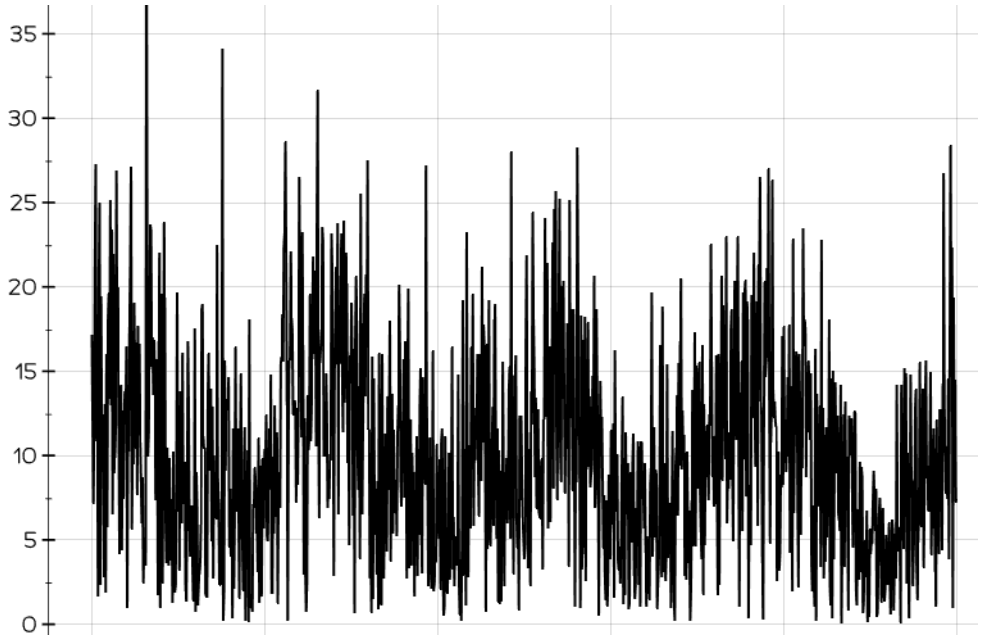
The applications of FFT are vast. It's used in audio compression like MP3s, for analysing brain waves in EEGs, and even in image processing to sharpen details or compress files.

4.4 Inverse Fourier Transform

The Inverse Fourier Transform (IFT) acts as the counterpart to the Fourier Transform (FT) within the realm of signal processing. While the FT decomposes a signal from the time domain (how it varies over time) into its constituent frequencies, the IFT performs the opposite task. It takes the frequency domain representation of a signal and reconstructs the original signal back into the time domain.Mathematically, the IFT is expressed as an integral that sums the contributions of all the frequency components present in the signal. Each frequency component is scaled by a factor and shifted by a complex exponential term. This summation essentially reverses the process of the FT, allowing us to recover the original signal from its frequency spectrum.

The IFT holds immense significance in signal processing for several reasons. Firstly, it enables us to analyze a signal's behavior in the frequency domain and then recreate the original signal for further processing or visualization in the time domain. This is crucial for tasks like denoising (removing unwanted frequencies) or feature extraction from signals.

Secondly, the IFT plays a vital role in various applications that involve transmitting or storing signals. Since the IFT allows us to reconstruct a signal from its frequency components, we can focus on transmitting or storing only the essential frequency information. This is particularly beneficial for data compression techniques like MP3 audio, where redundant frequency information is discarded before reconstruction using the IFT, leading to smaller file sizes.
Understanding the IFT is fundamental for effectively working with signals in the frequency domain. It empowers us to not only analyze and manipulate signals based on their frequency content but also to retrieve the original signal information for further use. The IFT serves as a cornerstone for various signal processing techniques, making it an indispensable tool in this vast and ever-evolving field.



Cleaned signal after FFT

4.5 Convolution

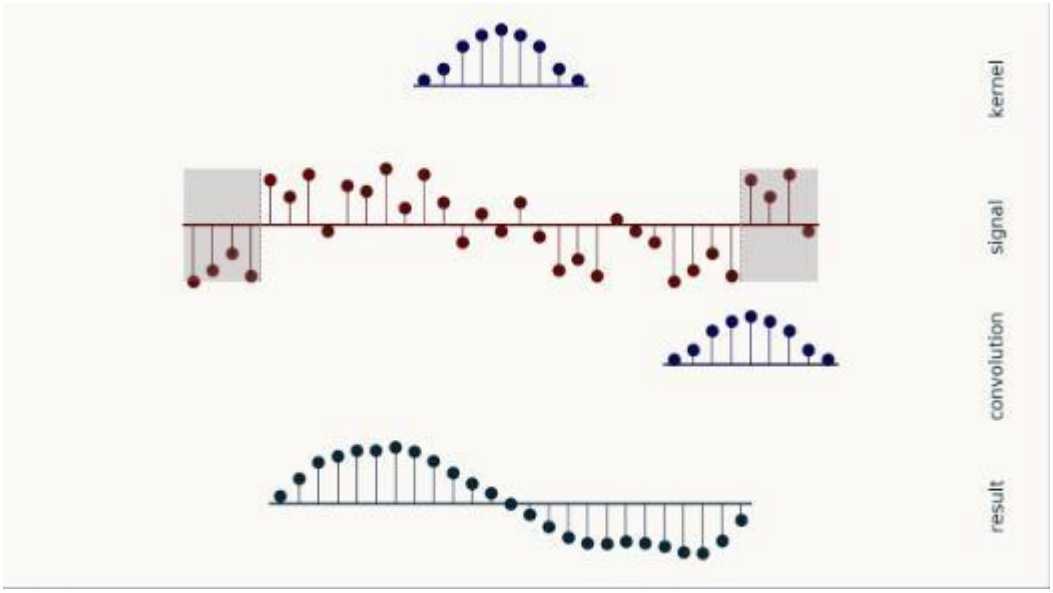
Convolution is a fundamental mathematical operation within signal processing. It acts on two functions, often denoted by f(t) and g(t), to produce a third function known as the convolution of f and g. This resulting function, typically written as f(t) * g(t), captures the overlapping influence of one function over the other as a function of a shift in position.

The mechanics of convolution involve a mathematical operation akin to a moving weighted average. We take one function, f(t), and reflect it across the y-axis. Then, we slide this reflected function, g(t), along the x-axis. At each position of this slide, we calculate the product of the two functions point-wise and integrate the product over the entire domain of one function (typically g(t)). This process effectively captures how much f(t) overlaps with the shifted version of g(t) at any given position. By repeating

this calculation for all possible shifts, we obtain the complete convolution function, f(t) * g(t).

Convolution finds numerous applications in signal processing due to its ability to describe how a system alters an input signal. For instance, in linear systems theory, the impulse response of a system, h(t), entirely characterizes its output when given any input signal, x(t). The convolution of the input signal with the impulse response, x(t) * h(t), yields the system's output signal, y(t). This demonstrates how convolution mathematically expresses the influence of a system on an input signal.

Beyond linear systems, convolution has applications in various signal processing tasks. It can be used for signal filtering by employing specific filter kernels as the second function in the convolution operation. Additionally, convolution plays a crucial role in image processing tasks like blurring and edge detection. By understanding convolution, we gain a powerful tool for analysing and manipulating signals across diverse domains within signal processing.



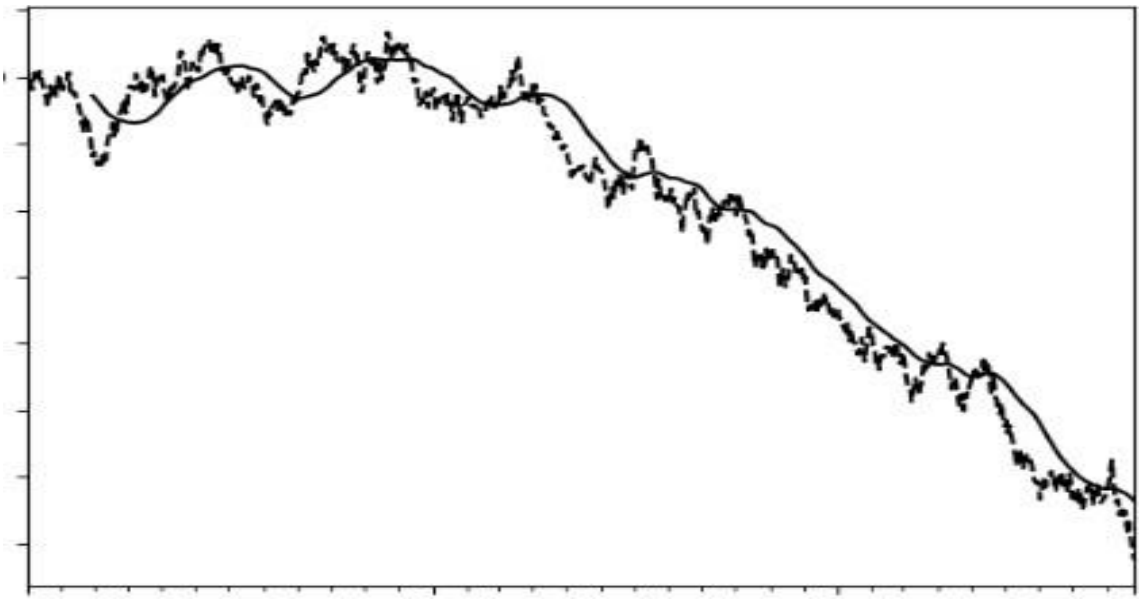
Example of convolution

4.6 Rolling Mean

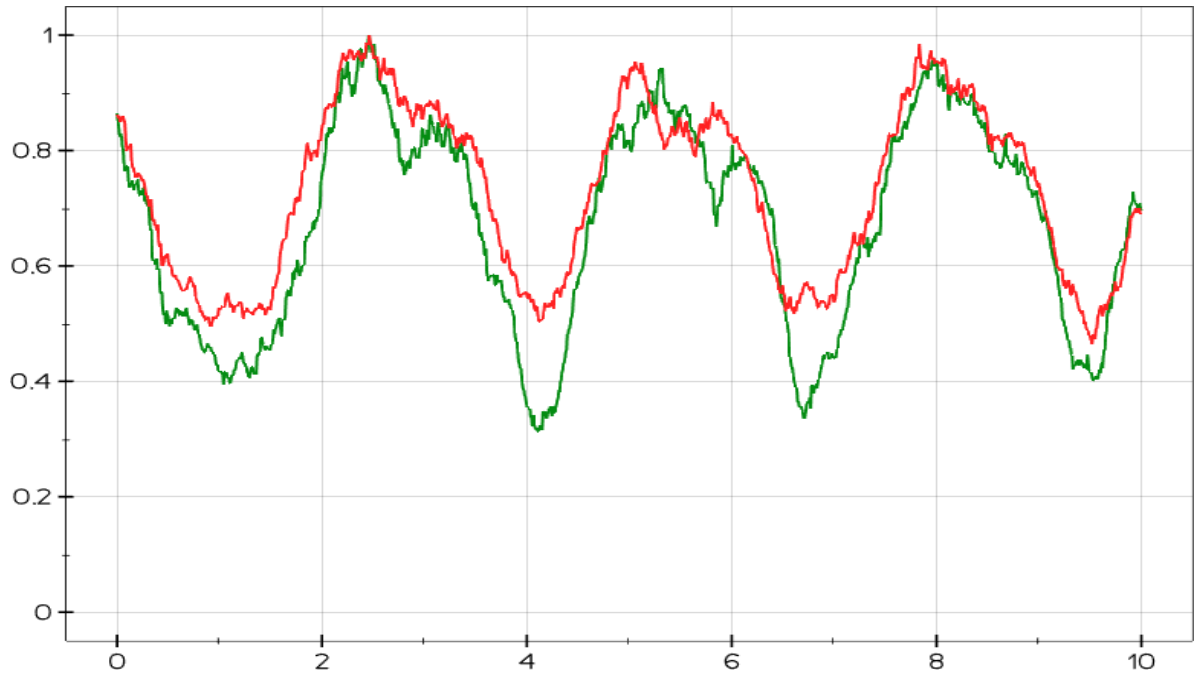
The rolling mean, also known as the moving average or running average, is a fundamental technique employed in signal processing for data smoothing and trend identification. It operates on a discrete-time signal, represented by a sequence of values {x(n)}, n = 0, 1, 2, ..., N-1. The rolling mean at a specific point n within the signal calculates the average of a predefined window of data points surrounding n. This window, often referred to as the window size (w), dictates the number of data points included in the average.

The calculation of the rolling mean at point n is achieved through a simple summation. We consider the w data points preceding n, the current point x(n) itself, and potentially w data points following n (depending on whether a centered or non-centered approach is used). The sum of these values is then divided by the total number of points involved (w + 1 for a centered mean, w for a non-centered mean). This process is repeated for each point n within the signal, resulting in a new sequence representing the rolling mean.

The impact of the window size on the rolling mean is crucial. A smaller window size leads to a more responsive mean, closely following rapid changes in the original signal. However, it may also capture and amplify high-frequency noise present in the data. Conversely, a larger window size provides a smoother mean, effectively suppressing noise but potentially lagging behind significant changes in the underlying trend of the signal. Selecting the optimal window size involves a trade-off between noise reduction and responsiveness, tailored to the specific characteristics of the signal and the desired outcome of the analysis.



Example of Rolling mean



Rolling Mean applied to our signal

4.7 Regression

In the realm of statistics and signal processing, regression emerges as a powerful tool for modeling the relationship between a dependent variable (what we're trying to predict) and one or more independent variables (what we're basing the prediction on). It essentially seeks to establish a mathematical equation that best fits the observed data points.

Imagine you're analyzing sales figures. The sales amount (dependent variable) might be influenced by factors like advertising spending (independent variable). Regression helps you discover the equation that best captures how changes in advertising spending translate to changes in sales. This equation allows you to predict future sales based on planned advertising budgets.

There are various regression techniques, with linear regression being the most fundamental. It aims to find a straight line that minimizes the overall difference between the predicted values from the equation and the actual data points. More complex regression models can capture non-linear relationships or involve multiple independent variables.

The applications of regression in signal processing are vast. It can be used for:

- **Signal Denoising**: By modeling the noise component in a signal as a separate variable, regression can help remove it, leaving behind the clean underlying signal.

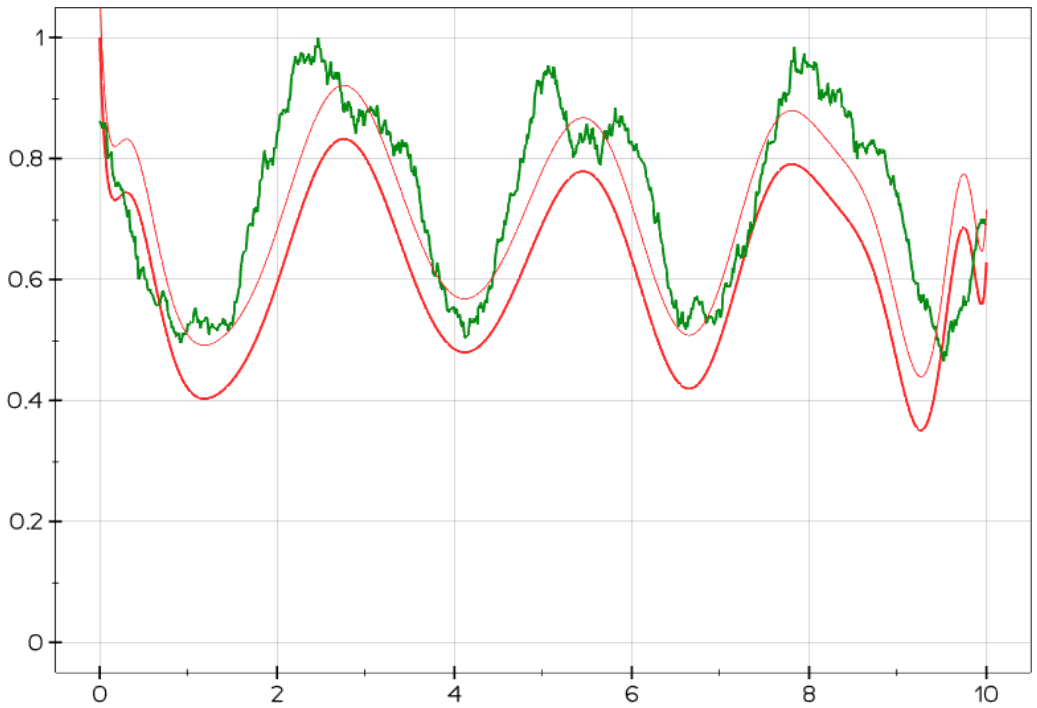
- **System Identification**: In linear systems, regression can be used to identify the mathematical model that describes how a system transforms an input signal into an output signal.
- **Signal Prediction**: Once a regression model captures the relationship between a signal and its influencing factors, it can be used to predict future values of the signal based on known or forecasted values of the independent variables.

Regression provides a powerful framework for understanding and manipulating signals. By establishing relationships between variables, it allows us to extract meaningful information, predict future behavior, and ultimately gain deeper insights from the data we analyze.

7.1 Polynomial Regression

When the relationship between variables in your data isn't neatly represented by a straight line, polynomial regression comes to the rescue. It's a technique used in statistics and signal processing to model non-linear relationships between a dependent variable (what you're trying to predict) and an independent variable (what you're basing the prediction on).

Here's the core idea: instead of fitting a straight line, polynomial regression uses a polynomial equation, which can curve and bend to better capture the trends in your data. The degree of the polynomial determines its complexity. A linear equation (degree 1) is essentially a straight line, while higher degrees (quadratic, cubic, etc.) allow for more intricate curves.



Polynomial Regression on our data

Imagine analyzing temperature data over time. A linear model might not capture the daily fluctuations, but a quadratic model could account for the highs and lows throughout the day. By increasing the polynomial degree, you can potentially fit the data even more closely, but there's a catch.

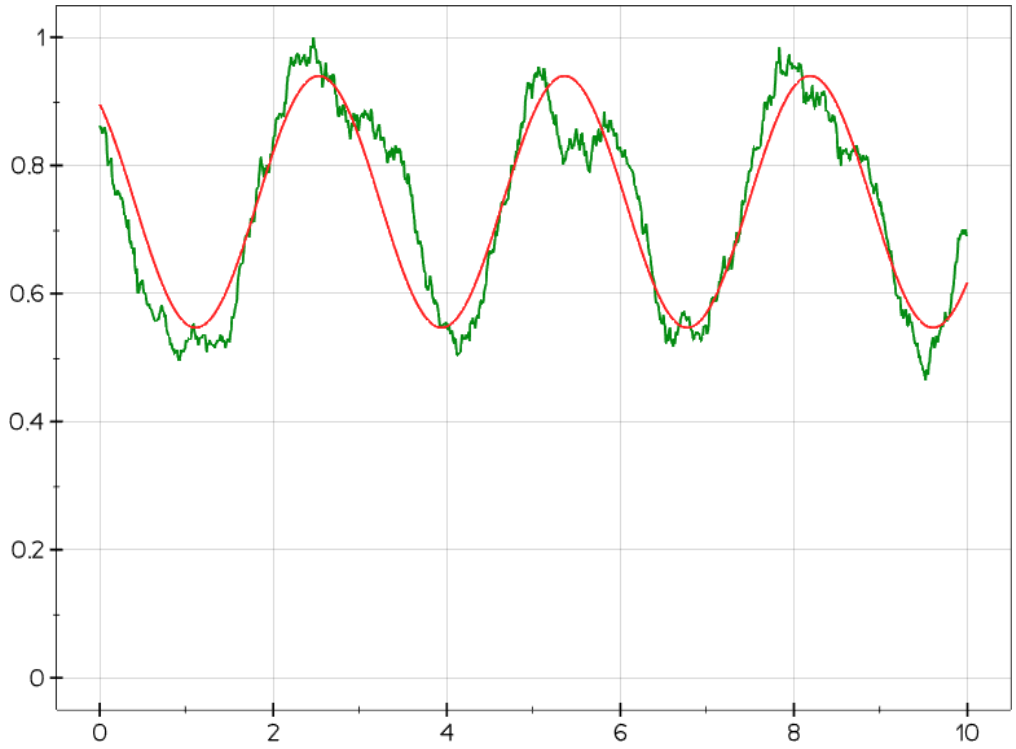
While a higher-degree polynomial might seem ideal for capturing every wiggle in your data, it can also lead to overfitting. This occurs when the model memorizes the specific data points too closely, losing its ability to generalize and make accurate predictions for unseen data. Therefore, choosing the right polynomial degree involves a balancing act between capturing the true trend and avoiding overfitting.

7.2 Sinusoid Fitting

Sinusoidal regression, also known as sinusoidal fitting, steps into the spotlight when your data exhibits a rhythmic or wave-like pattern. It's a technique employed in signal processing to find the best-fitting sinusoidal curve that captures the periodic fluctuations within your data.

Imagine you're analyzing temperature readings throughout the day. Unlike linear regression with its straight lines, sinusoidal regression allows you to model the cyclical rise and fall of temperatures over a 24-hour period. The resulting equation captures the amplitude (height of the wave), frequency (number of cycles within a specific time period), and any phase shift (horizontal movement of the wave) present in the data.

The core mathematical model behind sinusoidal regression involves a sine function with parameters like amplitude, frequency, phase shift, and a constant offset term. The goal is to adjust these parameters to minimize the difference between the fitted curve and the actual data points.



Sinusoidal Regression

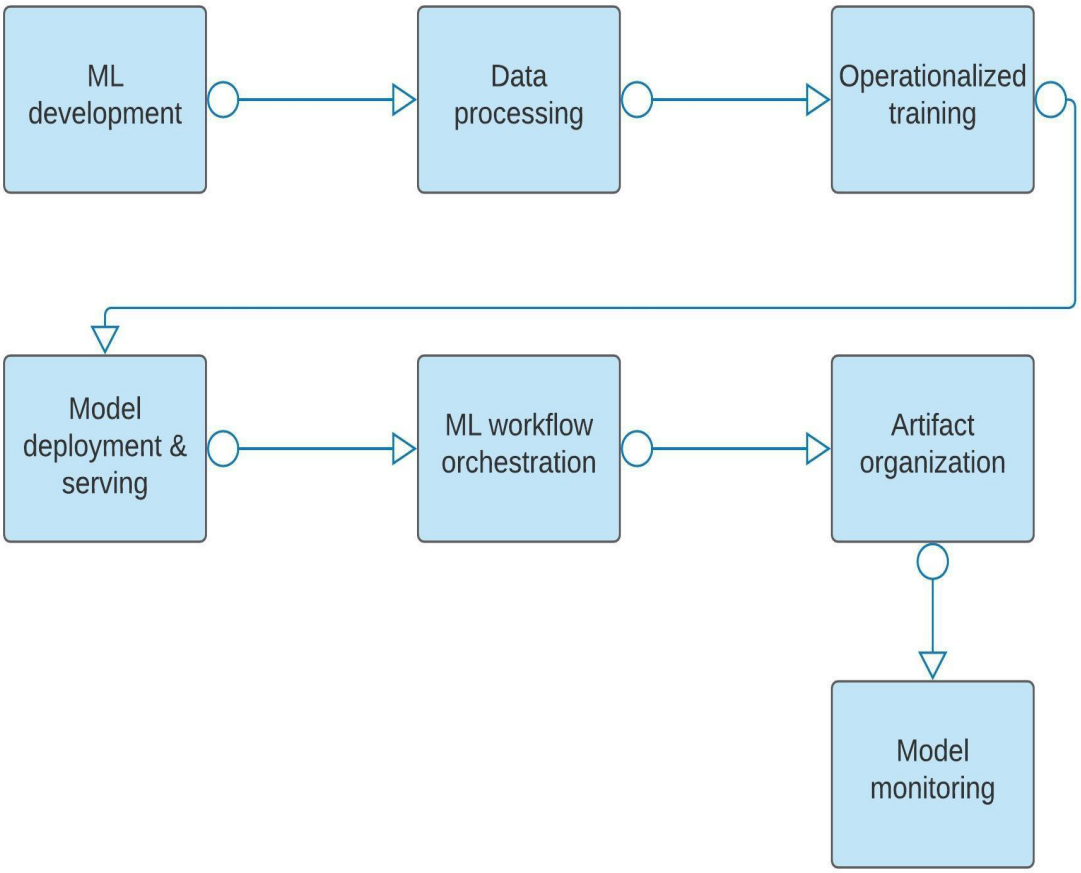
Here's what makes sinusoidal regression valuable:

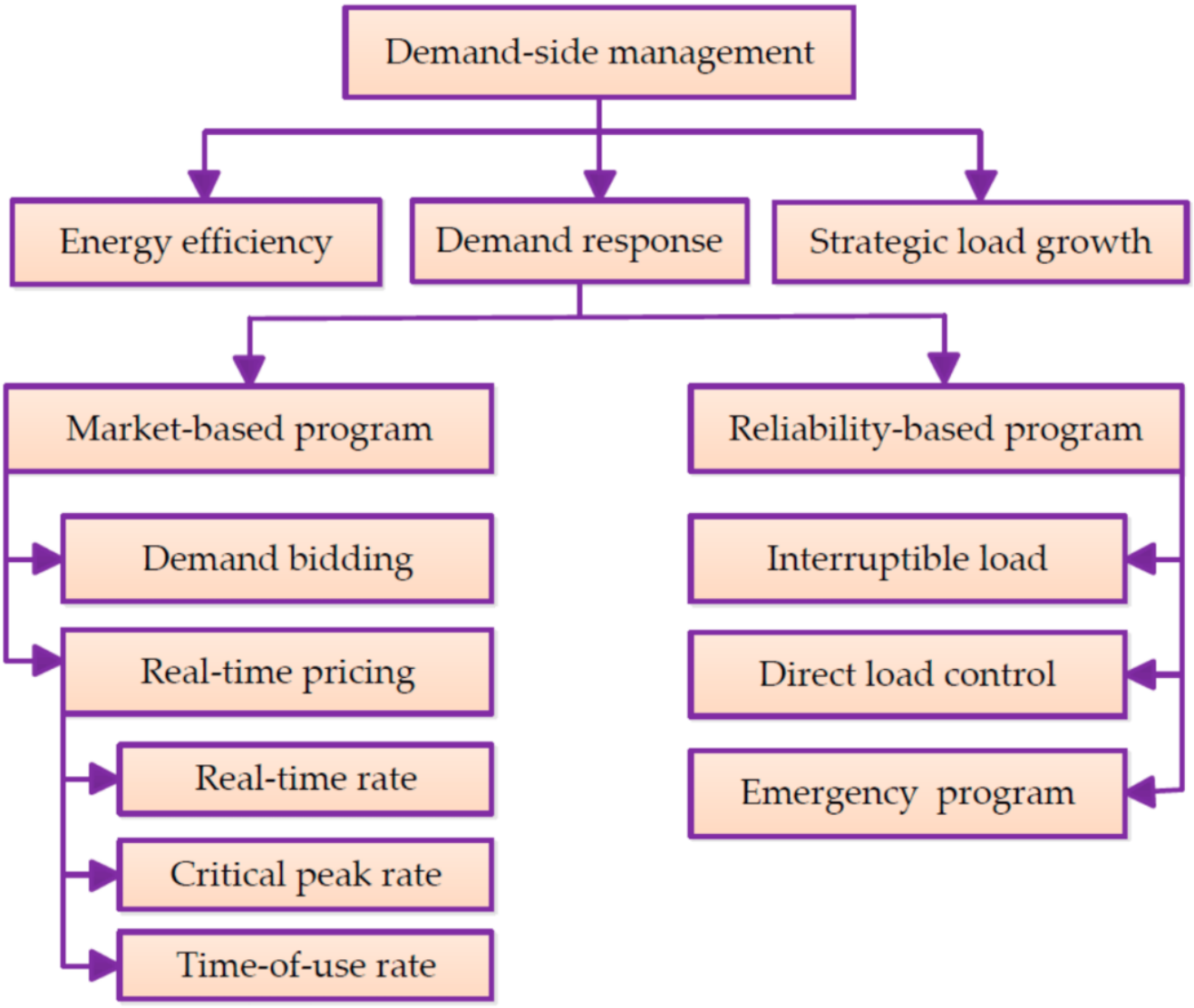
- **Modeling Periodic Signals:** It excels at analyzing data with inherent periodicity, such as sound waves, vibration patterns, or even economic cycles. By capturing the underlying sinusoidal nature of the signal, it allows for better understanding and prediction of its behavior.
- **Signal Filtering:** By identifying the frequency components present in a signal, sinusoidal regression can be used to isolate specific frequencies and remove unwanted noise that might be obscuring the desired signal.
- **Signal Prediction:** Once a sinusoidal model accurately captures the periodic behavior, it can be used to predict future values of the signal based on the estimated parameters. This can be helpful in tasks like forecasting seasonal trends or predicting equipment vibrations that might indicate potential issues.

CHAPTER 5.

CONCLUSION AND FUTURE WORK

5.1 Conclusion

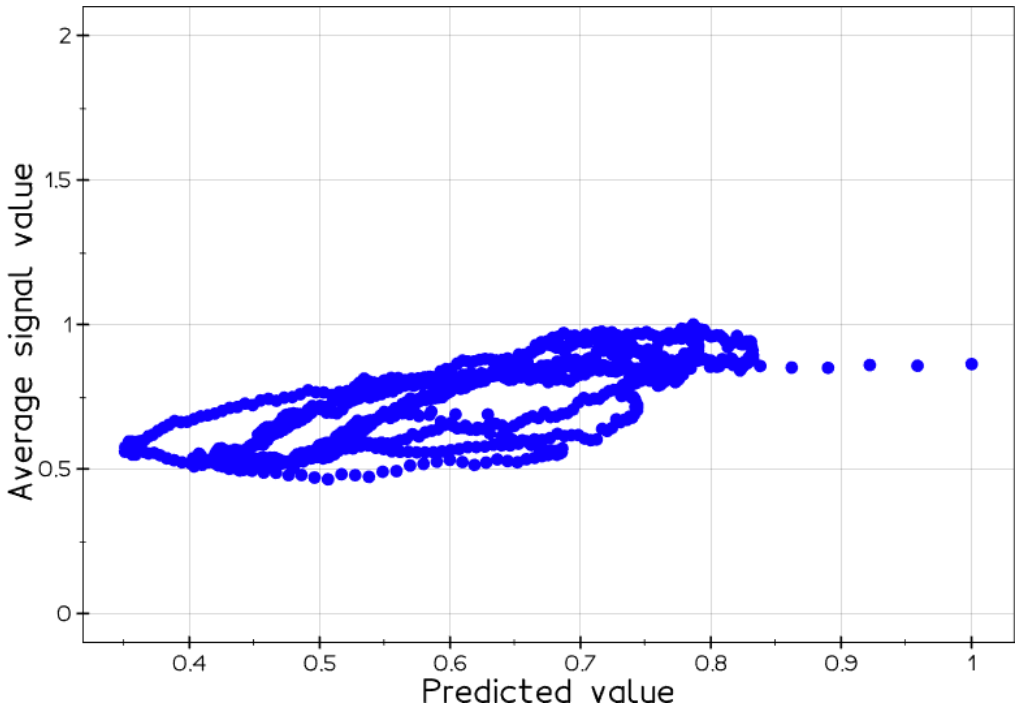




This project explored the potential of signal processing techniques for predicting demand-side electricity consumption. By leveraging techniques like e.g., Fast Fourier Transform, polynomial regression, sinusoidal fitting, we were able to develop a model that can forecast future electricity consumption patterns. The results demonstrate the effectiveness of signal processing in extracting valuable insights from historical consumption data.

We employed various signal processing techniques in our approach. The Fast Fourier Transform (FFT) was particularly useful for breaking down historical consumption data into its basic frequency components. This decomposition helped us identify hidden patterns and regular fluctuations within the data, revealing factors that influence how electricity demand changes over time. Additionally, we utilized techniques like polynomial regression and sinusoidal fitting. Polynomial regression was used to model non-linear relationships between consumption and influencing factors, such as weather patterns. Sinusoidal fitting, on the other hand, excelled at capturing the natural cyclical nature of electricity demand, specifically daily and seasonal variations. By combining these techniques, we were able to develop a robust

model capable of forecasting future electricity consumption patterns with greater accuracy.



Final relation between original and predicted values

The success of this project demonstrates the effectiveness of signal processing in extracting valuable insights from historical electricity consumption data. These insights allow us to make more accurate predictions of future demand. This improved prediction capability offers

significant benefits for various stakeholders in the energy sector. Utility companies and grid operators can leverage these predictions to anticipate peak demand periods, optimize their energy generation and distribution strategies, and potentially implement demand response programs. For individual consumers, the model could be a valuable tool for identifying patterns in their own consumption habits. This empowers them to make informed decisions about their energy usage, potentially leading to cost savings and a more sustainable energy footprint.

```
    for i in a:
        print("rolling mean ->", output2[i] / np.max(output2) )
        print("optimized curve ->", y[i])
        print("sinusoid curve is ->", optimized_curve[i])
```

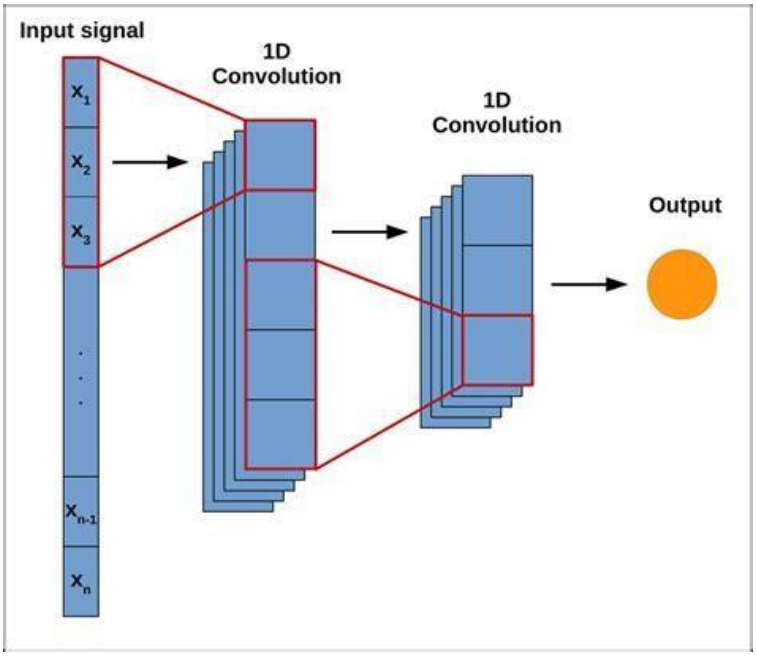
```
rolling mean -> 0.6490467980139836
optimized curve -> 0.6491874990598964
sinusoid curve is -> 0.663702726764576
rolling mean -> 0.9580142120329352
optimized curve -> 0.771266565374536
sinusoid curve is -> 0.9337975302260281
rolling mean -> 0.526605342121285
optimized curve -> 0.42234807791521317
sinusoid curve is -> 0.594546826466158
rolling mean -> 0.8412426205655331
optimized curve -> 0.7745302354453018
sinusoid curve is -> 0.9233412549200534
rolling mean -> 0.5489504851378759
optimized curve -> 0.4906827830462119
```

5.2Future Work

The world of signal processing is on a continuous journey of discovery, constantly innovating and expanding its capabilities. Here are some exciting areas where future work holds immense potential:

1. Convolutional Neural Networks (CNNs):

- **Stars in Image and Time-Series Data:** CNNs excel at processing data with a grid-like structure, making them ideal for tasks involving images and time-series signals (common in electricity consumption data).



- **Automatic Feature Extraction:**

  Unlike traditional methods that require hand-crafted features, CNNs automatically learn the most relevant features directly from the data. This is particularly beneficial for complex signals where feature extraction can be challenging.

- **Applications in Electricity Consumption:**

  CNNs can be used for tasks like:

  Demand forecasting: Unlike traditional methods that rely on hand-crafted features, CNNs automatically learn the most relevant patterns directly from the

data. This is crucial for electricity consumption data, which can be complex and exhibit subtle seasonal or weather-related variations. CNNs can identify these patterns and use them to predict future demand more effectively.

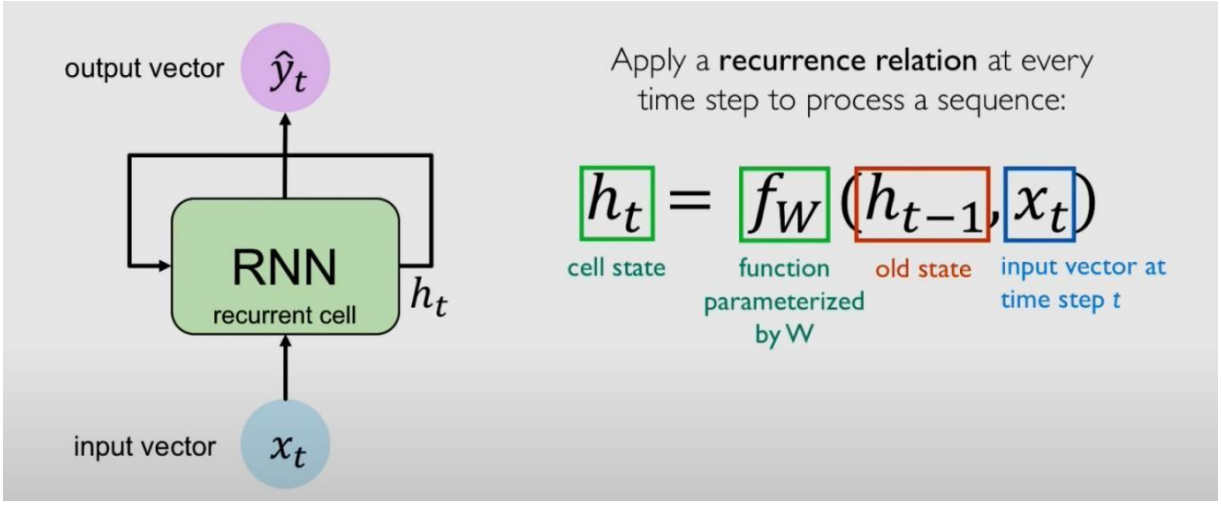
Anomaly detection: By training on historical data, CNNs develop an understanding of "normal" consumption patterns for specific consumers, locations, or equipment. The CNN continuously monitors new consumption data. When the data deviates significantly from the learned normal patterns, the model flags it as an anomaly.

Customer segmentation: By analyzing historical consumption data, CNNs can automatically group customers with similar consumption characteristics. This clustering helps identify different customer segments with distinct usage patterns. Once customer segments are identified, utilities can develop targeted energy-saving programs. For example, customers with high baseline consumption might benefit from energy efficiency audits, while those with predictable usage patterns might be ideal candidates for time-based pricing plans.

2. **Recurrent Neural Networks**

Capturing Sequential Information: RNNs are specifically designed to handle sequential data, where the order of information matters. This makes them well-suited for analyzing time-series data like electricity consumption, which exhibits temporal dependencies.

Long Short-Term Memory (LSTM) Networks: A powerful variant of RNNs, LSTMs are adept at capturing long-term dependencies within sequences. This is crucial for electricity consumption data, as historical patterns can influence future demand.



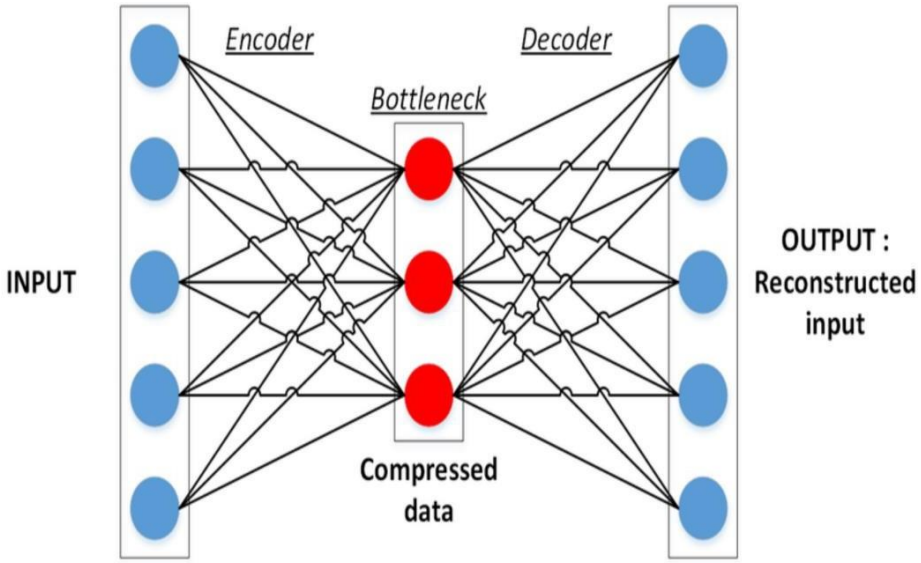
RNNs can be used for:

- **Demand forecasting:** LSTMs can consider historical trends and seasonal variations to make more accurate predictions.

- **Short-term load forecasting**: They can predict demand fluctuations within a shorter timeframe, useful for grid operators to balance supply and demand in real-time.
- **Demand response program optimization:** By analyzing customer response patterns, LSTMs can help design more effective demand response programs to incentivize energy conservation during peak hours.

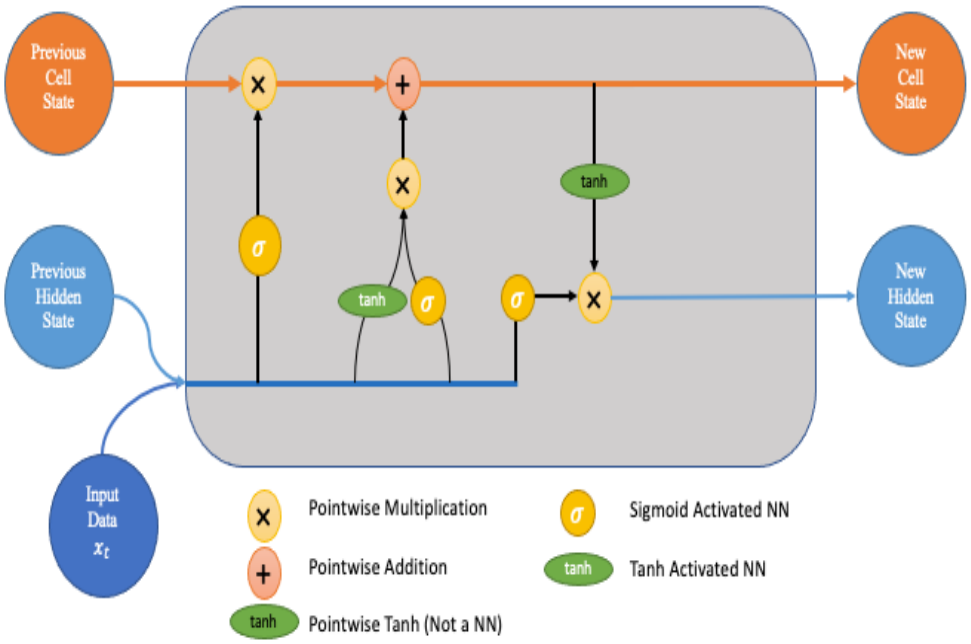
3. **Autoencoders:**

- **Dimensionality Reduction and Anomaly Detection**: Autoencoders are a special type of neural network that learns to compress data into a lower-dimensional representation while attempting to reconstruct the original data from the compressed version.
- **Identifying Unusual Patterns**: The reconstruction error can be used to identify anomalies in the data. This is valuable for detecting unusual consumption patterns that might indicate fraudulent activity or equipment failures.



- **Applications in Electricity Consumption**: Autoencoders can be used for:
  Anomaly detection: Identifying unusual consumption patterns that deviate significantly from the learned compressed representation.
  Data denoising: Autoencoders can be trained to remove noise from consumption data, improving the quality of the signal for further analysis.
  Feature extraction: The learned compressed representation can be used as a new set of features for classification or prediction tasks.

4. LSTM:



Long Short-Term Memory (LSTM) is a specialized type of recurrent neural network (RNN) architecture tailored to address the challenges of learning long-term dependencies in sequential data. LSTMs excel in tasks like time-series prediction and natural language processing, thanks to their ability to retain relevant information over extended sequences.

Comprising essential elements like the cell state, input gate, forget gate, and output gate, an LSTM unit manages memory, selectively updates information, discards irrelevant data, and generates outputs based on current input and past memory. This sophisticated architecture allows LSTMs to effectively capture complex patterns and dependencies in sequential data.

With their capability to learn and remember patterns over long sequences, LSTM networks have become indispensable in various fields, from machine translation and sentiment analysis to speech recognition and time-series forecasting.

The key components of an LSTM unit include:

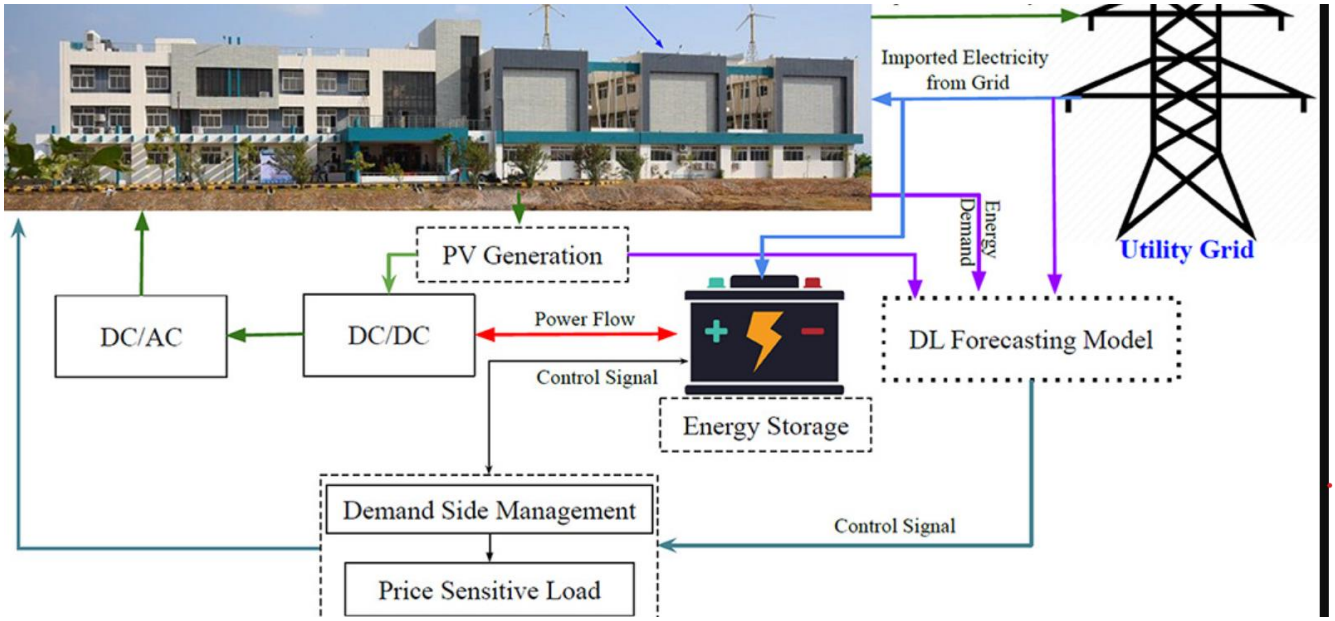
Cell State: This represents the memory of the network and allows information to persist over long sequences. The cell state can be updated or modified through various operations, ensuring relevant information is retained and irrelevant information is discarded.

Input Gate: Determines how much new information is added to the cell state at each time step. It selectively updates the cell state based on the current input and the past memory.

Forget Gate: Controls which information from the previous cell state should be discarded or forgotten. It decides which information is no longer relevant for the current prediction task.

Output Gate: Determines the output of the LSTM unit. It selectively outputs information from the cell state based on the current input and the updated cell state.

Overall, LSTM networks have become a cornerstone in deep learning for sequential data processing and have been instrumental in achieving state-of-the-art performance in various applications, including machine translation, speech recognition, sentiment analysis, and time-series forecasting.


Imported Electricity
from Grid
Energy
Demand
Utility Grid
PV Generation
DC/AC
DC/DC
Power Flow
Control Signal
Energy Storage
DL Forecasting Model
Demand Side Management
Price Sensitive Load
Control Signal

REFERENCES

1. Power consumption prediction in urban areas using Machine Learning as a strategy towards smart cities. AJBAR Vol1 (1), 2-22: 11-24, ISSN: 2811-2881
2. An overview of electricity demand forecasting techniques, Vol.3, No.2, 2013-National Conference on emerging trends in electrical, instrumental and communication engineering.
3. A deep learning architecture for power management in smart cities. Qin Xin a, Mamoun Alazab b, Vincente Garcia Diaz c, Carlos Enrique Montenegro-Marin d, Ruben Gonzalez Crespo e.
4. Strategies for predictive power: Machine Learning models in city-scale load forecasting, e-Prime-Advances in Electrical Engineering, Electronics and Energy, Vol.6 December 1023, 1—392
5. Machine Learning for modern power distribution systems: Progress and perspectives. Marija Markovic, Matthew Bossart, Bri-Mathias Hodge. Journal of Renewable and Sustainable energy, Vol. 15, Issued 3 May 2023.
6. Trees vs Neurons: Comparison between random forest and ANN for high-resolution prediction of building energy consumption. Ahmad, M. W., Mourshed, M., & Rezgui, Y. (2017). Energy and buildings, 147, 77-89.
7. Data driven prediction models of energy use of appliances in a low-energy house. Energy and buildings, 140, 81-97, Candanedo, L. M., Feldheim, V., & Deramaix, D. (2017).
8. Forecasting the Usage of Household Appliances Through Power Meter Sensors for Demand Management in the Smart Grid, A.Barbato, A.Capone, M. Rodolfi, D. Tagliaferri Dipartimento di Elettronica e Informazione Politecnico di Milano, Italy.
9. A review on artificial intelligence-based load demand forecasting techniques for smart grid and buildings, Renew. Sustain. Energy Rev.50 (2015) 1352–1372, M.Q. Raza, A. Khosravi,
10. Time series forecasting for decision making on city-wide energy demand: a comparative study, in: 2022 International Conference on Decision Aid Sciences and Applications, DASA 2022, 2022, pp. 1706–1710, O. Nooruldeen, S. Alturki, M.R. Baker, A. Ghareeb,
11. A comprehensive survey on machine learning-based big data analytics for IoT-enabled smart healthcare system, Mob. Netw. Appl. 26 (1) (2021) 234–25, W. Li.
12. Regression Based Peak Load Forecasting Using a Transformation Technique, IEEE Transaction on Power System, Vol.9, pp.1788–1794, 1994, T. Haida and S. Muto.
13. Short Term Load Forecasting. Proceedings of the IEEE, Vol.75, pp. 1558-1573, 1987,]. G. Gross G. Gross, and F. D. Galiana,
14. Uncertainty management in electricity demand forecasting with machine learning and ensemble learning: case studies of COVID-19 in the US metropolitans, Eng. Appl. Artif. Intell. 123 (2023), 106350, M.R. Baker.