

Naan Mudhalvan
IBM Professional Readiness Program

Date	20 May 2023
Team ID	NM2023TMID21277
Project Name	Project - A Reliable Energy Consumption Analysis System For Energy-Efficient Appliances

PBL-NT-GP--7992-1681102296

**Project - A Reliable Energy
Consumption Analysis System For
Energy-Efficient Appliances**

Project Report Document

TABLE OF CONTENTS

1	INTRODUCTION	
	1.1 Project Overview	
	1.2 Purpose	
2	IDEATION & PROPOSED SOLUTION	
	2.1 Problem Statement Definition	
	2.2 Empathy Map Canvas	
	2.3 Ideation & Brainstorming	
	2.4 Proposed Solution	
3	REQUIREMENT ANALYSIS	
	3.1 Functional requirement	
	3.2 Non-Functional requirements	
4	PROJECT DESIGN	
	4.1 Data Flow Diagrams	
	4.2 Solution & Technical Architecture	
	4.3 User Stories	
5	CODING & SOLUTIONING (Explain the features added in the project along with code)	
	5.1 Feature 1	
	5.2 Feature 2	
6	RESULTS	
	6.1 Performance Metrics	
7	ADVANTAGES & DISADVANTAGES	
8	CONCLUSION	
9	FUTURE SCOPE	

Source Code

GitHub & Project Video Demo Link

1 INTRODUCTION

1.1 PROJECT OVERVIEW

Power consumption analysis for households is an ML project that involves using machine learning algorithms to analyse and predict the energy consumption patterns of residential buildings. The goal of this project is to help homeowners and utility companies better manage their energy usage, reduce waste, and lower costs.

The project involves collecting data on energy consumption and related factors. This data is then used to train machine learning models to make accurate predictions of future energy consumption based on these factors. The models can be used to identify patterns in energy usage and make recommendations for ways to reduce energy waste and improve efficiency.

Overall, power consumption analysis for households is an important application of machine learning that has the potential to make a significant impact on energy usage and sustainability.

1.2 PURPOSE

The following points gives the main purpose of this project:

1. **Energy Usage Understanding:** The project aims to analyze and understand the energy consumption patterns of residential buildings. By collecting data on various parameters such as active power, reactive power, voltage, and current intensity, the project seeks to gain insights into how energy is being utilized within households.
2. **Prediction of Future Consumption:** Machine learning algorithms are applied to the collected data to train models that can accurately predict future energy consumption based on historical patterns. These predictions can help homeowners and utility companies plan and manage their energy usage more effectively.
3. **Efficient Energy Management:** By analyzing energy sub-metering data from different areas of the house, such as the kitchen, laundry room, and electric

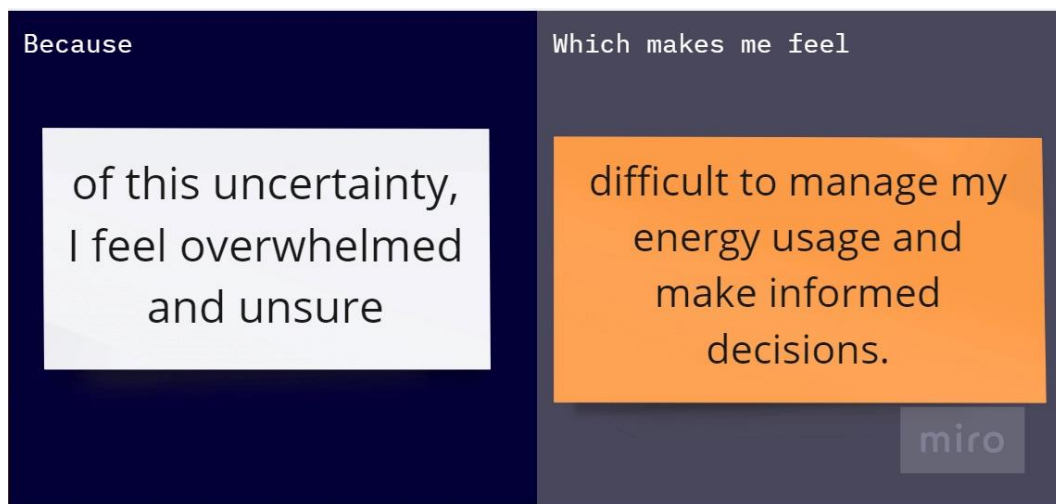
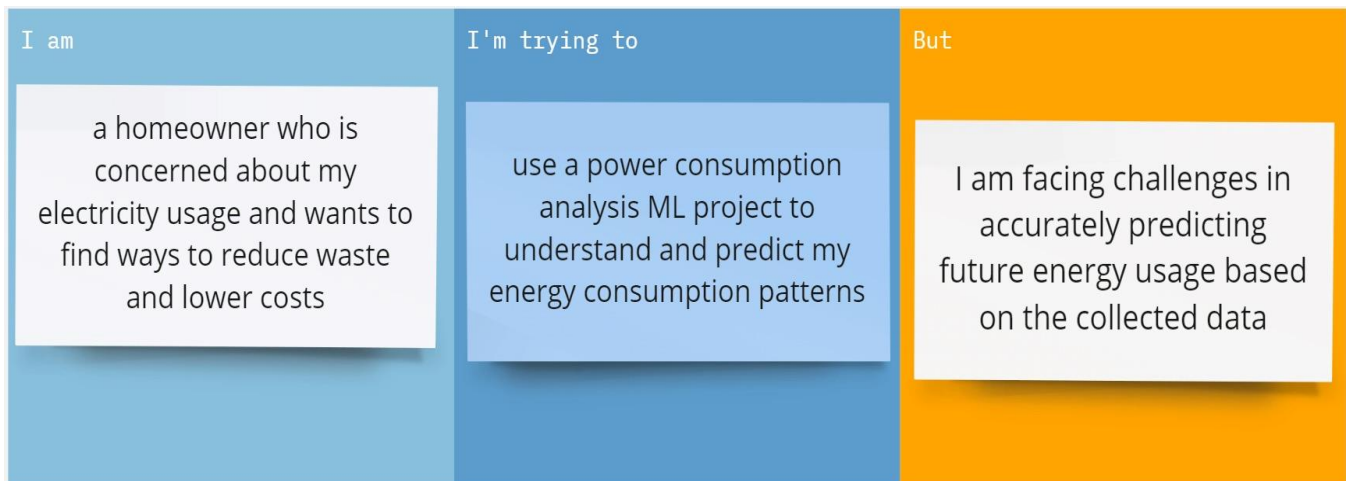
water-heater, the project aims to identify areas of high energy consumption and waste. This information can be used to suggest strategies for reducing energy waste and improving efficiency in specific areas of the household.

4. **Cost Reduction:** By understanding energy consumption patterns and identifying areas of inefficiency, the project aims to help homeowners and utility companies lower their energy costs. The insights gained from the analysis can guide decisions on optimizing energy usage, which may lead to significant cost savings over time.
5. **Sustainability and Environmental Impact:** By promoting efficient energy usage and waste reduction, the project contributes to sustainability efforts and reduces the environmental impact associated with energy production. By empowering homeowners and utility companies with information and recommendations, the project helps promote a more sustainable and greener approach to energy consumption.
6. **User-Friendly Recommendations:** The project intends to provide user-friendly recommendations and insights based on the analysis of energy consumption patterns. These recommendations can be customized to the specific needs and preferences of homeowners, making it easier for them to implement energy-saving measures and reduce waste.
7. **Collaboration with Utility Companies:** The project aims to establish collaboration with utility companies, allowing them to leverage the analysis and predictions to optimize energy distribution and management. By sharing insights and data, the project can contribute to the development of more efficient and sustainable energy grids and infrastructure.

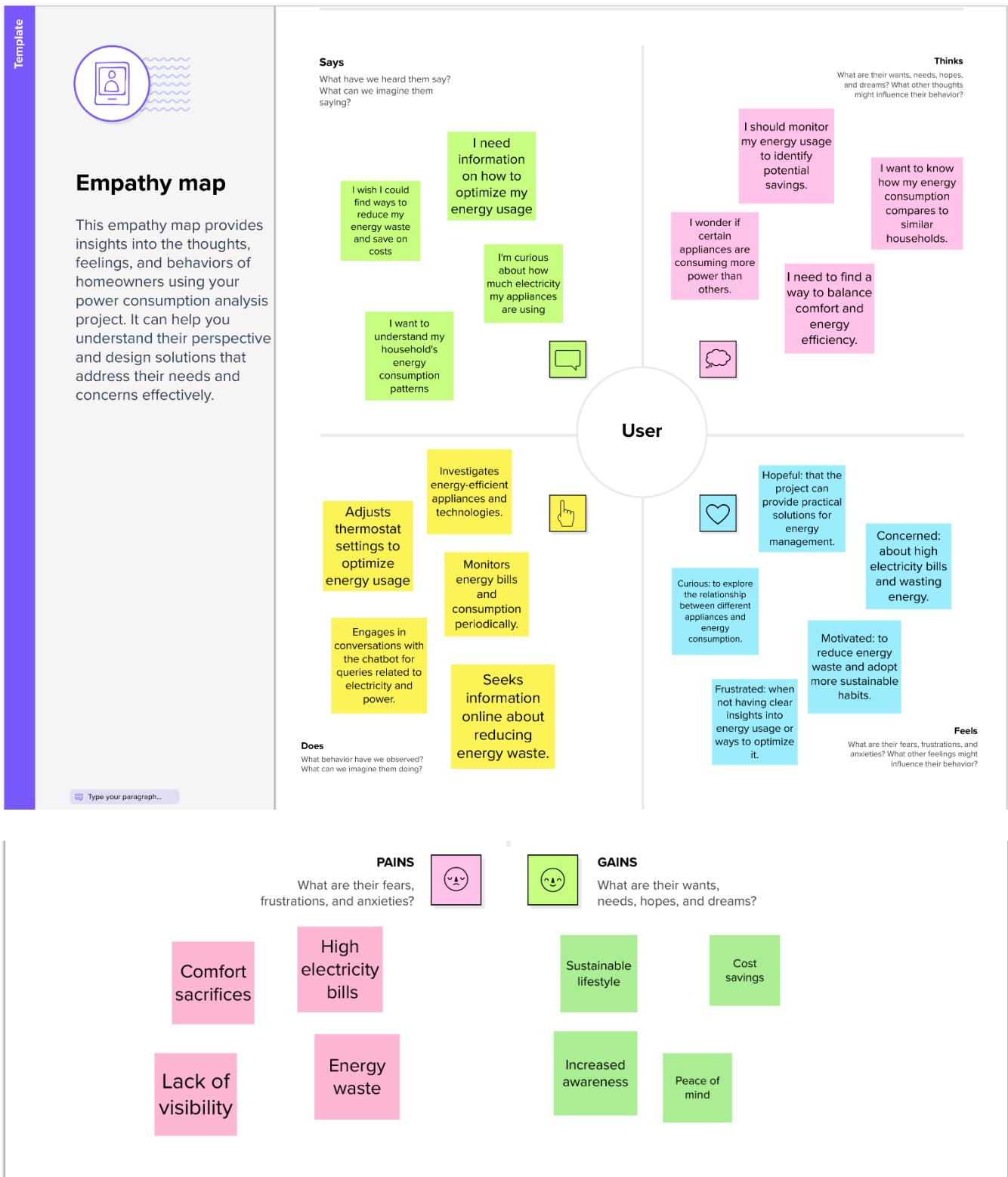
2 IDEATION & PROPOSED SOLUTION

2.1 Problem Statement Definition

I am a homeowner who is concerned about my electricity usage and wants to find ways to reduce waste and lower costs. I am trying to use a power consumption analysis ML project to understand and predict my energy consumption patterns. However, I am facing challenges in accurately predicting future energy usage based on the collected data. Because of this uncertainty, I feel overwhelmed and unsure about how to effectively manage my energy usage and make informed decisions.



2.2 Empathy Map Canvas



2.3 Ideation & Brainstorming

S1: Defining problem statement:



Brainstorm & idea prioritization

Brainstorming Session for the project on Power Consumption Analysis



Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕒 10 minutes

A Team gathering

Suriyaa V
Kartikeyan TR
Srikantan U
Hamilton Samic S

B Set the goal

Power Consumption Additional ideas

1

Define your problem statement

PROBLEM

Develop a power consumption analysis system for households using machine learning algorithms to predict energy consumption patterns.

S2: Brain Storm and Group ideas:

2

Brainstorm

Ideas coming to mind from the 4 members

Person 1

Help homeowners and utility companies manage their energy usage better.

Investigate the impact of weather conditions on energy usage and incorporate weather data into the analysis

Implement a chatbot to solve queries regarding electricity and power.

Person 2

Train machine learning models to make accurate predictions of future energy consumption.

Provide recommendations for reducing energy waste and improving efficiency.

Person 3

Collect data on energy consumption and related factors

Use machine learning algorithms to analyze and predict energy consumption patterns.

Person 4

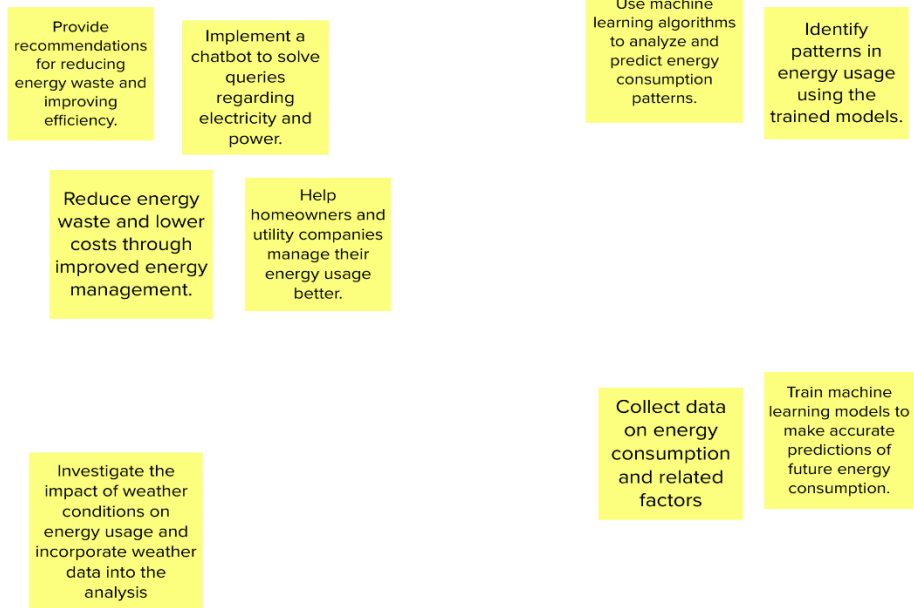
Identify patterns in energy usage using the trained models.

Reduce energy waste and lower costs through improved energy management.

3

Group ideas

Grouping of similar ideas

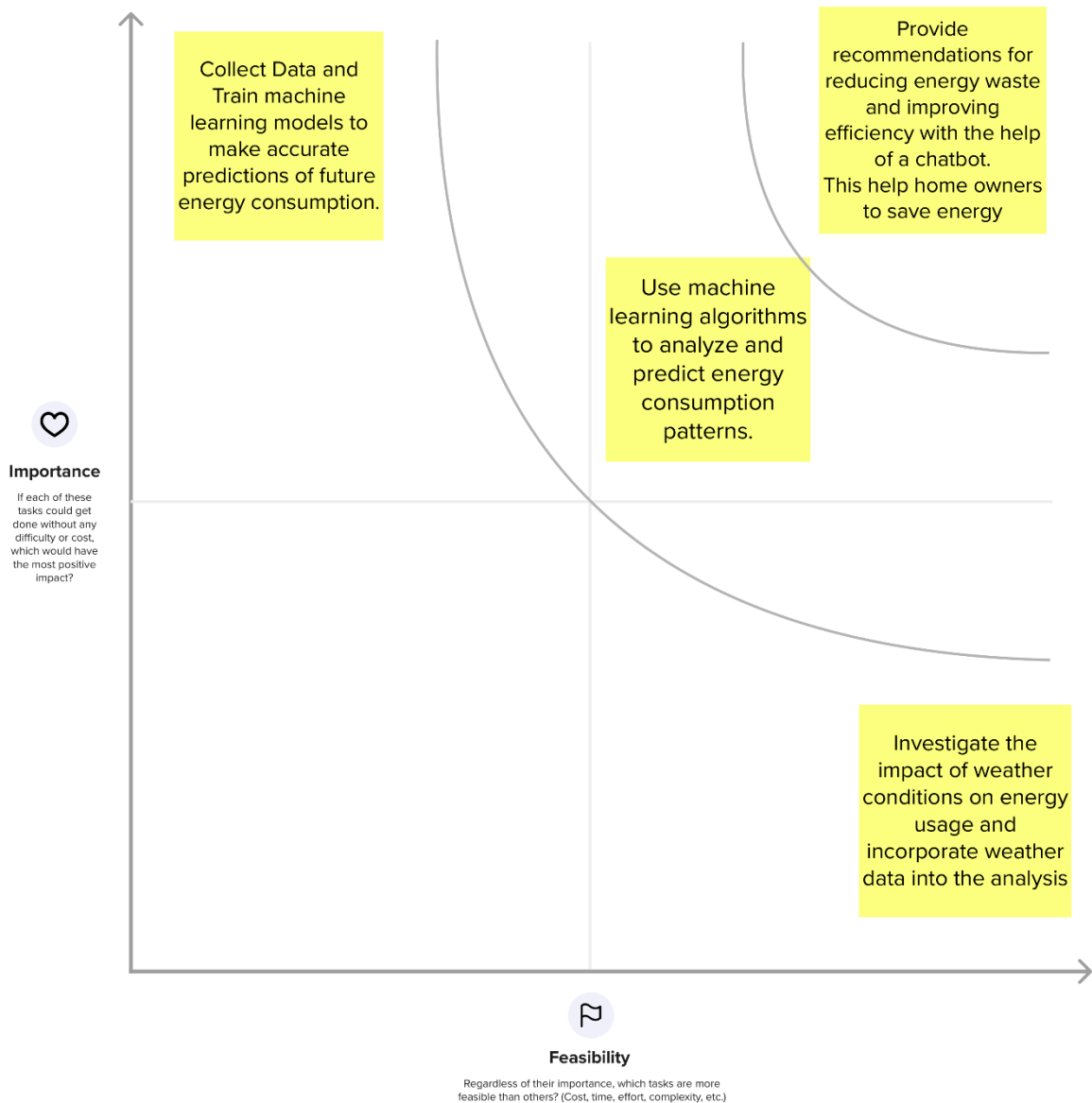


S3: Prioritization:

4

Prioritize

Ideas with different priorities are arranged in a graph



2.4 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To perform energy consumption analysis in order to identify the energy efficient appliances

2.	Idea / Solution description	Active world power prediction using Regression models. Also we have added additional novelties.
3.	Novelty / Uniqueness	<p>➔ Time series prediction of active power consumption using LSTM</p> <p>➔ Chatbot which answers our queries related to power and electricity</p>
4.	Social Impact / Customer Satisfaction	The proposed system promotes sustainable consumption, reduces carbon footprint, and contributes to a greener planet, while also providing insight on appliances
5.	Business Model (Revenue Model)	Our business model is based on offering a subscription-based energy consumption analysis system for energy-efficient appliances.
6.	Scalability of the Solution	The proposed system is designed to be scalable, accommodating increasing data volume and user demand, maintaining performance

3 REQUIREMENT ANALYSIS

3.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No	Functional Requirement	Sub Requirement
FR 1	Data collection	Collect and store data on energy consumption and related factors, such as global active power, global reactive power, voltage, and sub-metering.
FR 2	Data pre processing	Tasks such as data cleaning, handling missing values, outlier detection, and normalization. This ensures the data is in a suitable format for training and analysis.

FR 3	Training ML models	Training of different models should to learn the patterns and relationships between energy consumption and the associated factors.
FR 4	Prediction and forecasting	Predicting global active power and potentially performing time series analysis for forecasting energy usage over specific time periods, such as the next few months.
FR 5	Analysis and insights	Identifying peak demand periods, detecting seasonal variations, and uncovering correlations between energy usage and factors like weather conditions or occupancy patterns.
FR 6	Recommendations and optimization	Suggesting energy-saving practices, adjusting appliance usage schedules, or promoting the adoption of energy-efficient technologies.
FR 7	Visualization and reporting	Generating charts, graphs, and reports that allow users to easily interpret and gain insights from the energy consumption analysis.
FR 8	Query handling	Users should be able to ask questions, seek clarification, and receive appropriate responses based on the available information and analysis.
FR 9	Integration and scalability	Using of smart meters or weather APIs, to enhance the accuracy and coverage of the analysis. Additionally, the project should be scalable to handle larger datasets and accommodate future data growth.

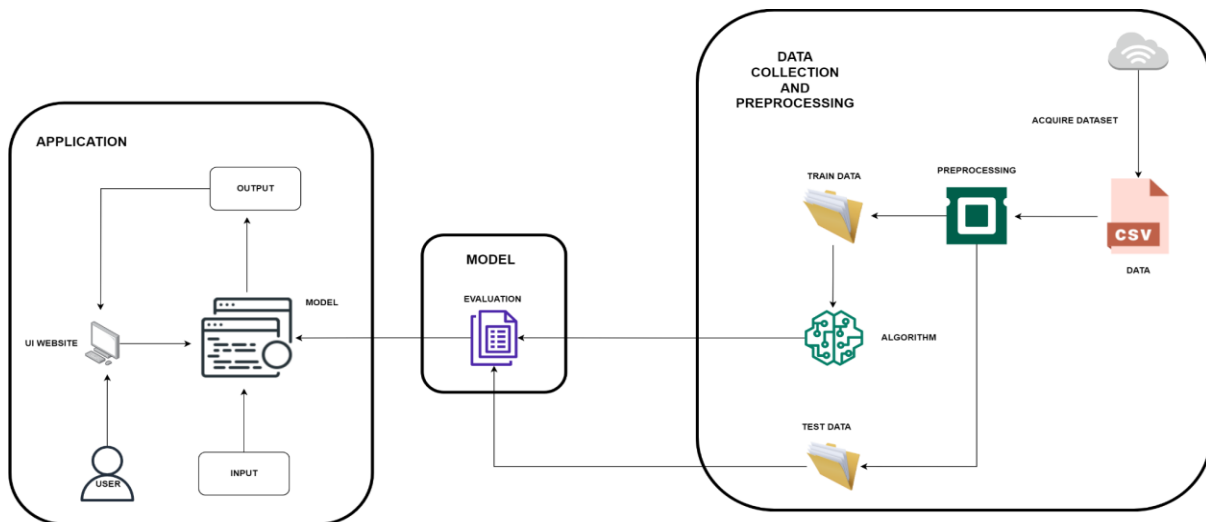
3.2 Non-Functional requirements

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The system has an intuitive user interface and user-friendly features for easy navigation and efficient interaction.
NFR-2	Security	Provides robust user authentication, and access control mechanisms to safeguard sensitive energy consumption data.
NFR-3	Reliability	The system consistently generates accurate energy consumption analysis and ensure minimal downtime or system failures
NFR-4	Performance	The system delivers fast responses and handles large data volumes efficiently to provide timely and efficient energy consumption analysis
NFR-5	Availability	Ensures high availability of the system and provides reliable access to energy consumption analysis capabilities.
NFR-6	Scalability	The system easily scales to accommodate growing data volumes and user demands without compromising performance or functionality.

4 PROJECT DESIGN

4.1 Data Flow Diagrams



4.2 Solution & Technical Architecture

Data Collection Layer:

Data Sources: The architecture should support the collection of data from various sources, such as smart meters, weather APIs, occupancy sensors, or utility companies' databases.

Data Collection Pipeline: Establish a robust pipeline to collect data at regular intervals, ensuring data integrity and reliability. This pipeline should handle data ingestion, data quality checks, and storage in a centralized database or data lake.

Data Preprocessing and Feature Engineering Layer:

Data Cleaning and Transformation: Implement preprocessing techniques to handle missing values, outliers, and inconsistencies in the collected data. This ensures the data is of high quality and suitable for further analysis.

Feature Engineering: Derive additional features from the raw data that can capture important patterns and relationships. This can involve creating lagged variables, aggregating data over different time intervals, or incorporating external factors like weather conditions.

Machine Learning Model Training and Deployment Layer:

Model Selection: Choose appropriate machine learning algorithms, such as regression models or time series forecasting models, based on the project requirements and data characteristics.

Model Training: Train the selected models using the preprocessed data. Utilize techniques like cross-validation and hyperparameter tuning to optimize the model's performance.

Model Deployment: Deploy the trained models in a production environment, ensuring scalability and efficiency. This can be achieved using platforms like cloud-based services or containerization technologies.

Prediction and Analysis Layer:

Real-time Prediction: Develop mechanisms to make real-time predictions of energy consumption based on the trained models. This can involve processing incoming data streams and applying the models to generate accurate and up-to-date predictions.

Energy Consumption Analysis: Analyze the predicted energy consumption patterns to identify peak demand periods, seasonality, and correlations with external factors. Generate insights and visualizations to help users understand the patterns and make informed decisions.

Recommendation and User Interface Layer:

Personalized Recommendations: Based on the analysis results, provide personalized recommendations to homeowners on optimizing energy usage and reducing waste. This can include suggestions for adjusting appliance usage schedules, adopting energy-saving practices, or leveraging renewable energy sources.

User Interface: Develop a user-friendly interface, such as a web or mobile application, to allow users to interact with the system. The interface should provide access to visualizations, reports, and query submission functionalities. It can also integrate a chatbot or AI-powered assistant to address user queries regarding electricity and power-related topics.

Integration and Scalability:

Integration with External Systems: Ensure seamless integration with external systems, such as smart home automation systems or utility company databases, to enhance data availability and accuracy.

Scalability: Design the architecture to handle large volumes of data and accommodate future growth. Utilize scalable storage and computing resources, such as cloud-based solutions, to support increased data processing demands.

Security and Privacy:

Data Security: Implement appropriate security measures to protect the collected data, including encryption, access controls, and user authentication mechanisms.

Privacy Considerations: Comply with privacy regulations and ensure the proper anonymization or aggregation of sensitive user data.

Solution Architecture Diagram:

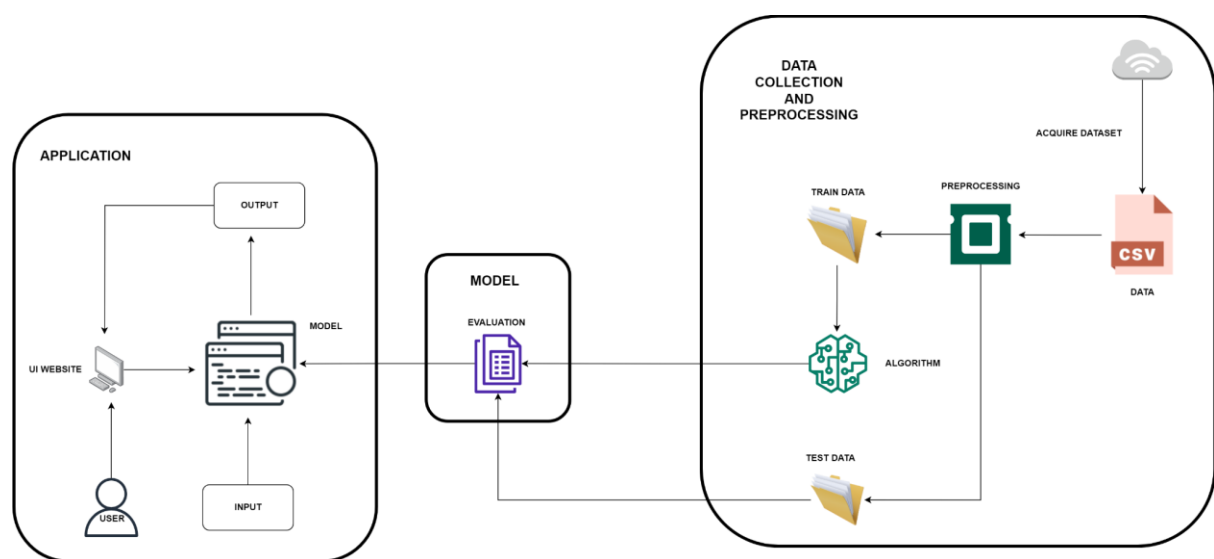


Figure 1: Architecture and data flow of the Power Prediction Application

4.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
Customer (Mobile user)	Data Visualization	USN-1	As a user, I want to view my energy consumption data in an interactive graph.	The graph displays energy consumption over time, allows zooming and panning, and shows data for different appliances.	High	Suriyaa V
		USN-2	As a user, I want to receive personalized energy-saving recommendations based on my consumption patterns	The recommendations suggest specific actions to reduce energy waste and improve efficiency.	High	Kartikeyan TR
		USN-3	As a user, I want to set energy usage goals and track my progress over time.	The system displays my current energy usage compared to my goals, providing insights into my consumption habits.	Medium	Suriyaa V
		USN-4	As a user, I want to receive notifications when my energy consumption exceeds a predefined threshold.	The system sends me alerts when my consumption goes above the set limit, helping me stay aware and take necessary actions	Medium	Srikantan
		USN-5	As a user, I want to analyze my energy consumption patterns using advanced statistical models	The system provides statistical analysis, including trends, seasonality, and correlation with external factors, to help me understand my usage patterns better	High	Suriyaa
		USN-6	I want to compare my energy consumption with similar households in my area.	The system provides benchmarking features, allowing me to see how my usage compares to similar households and identify areas for improvement.	Medium	Kartikeyan
		USN-7	I want access to customer energy consumption data and usage patterns.	The system provides a dashboard with customer information and historical usage data for efficient	Low	Hamilton

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
				customer support and troubleshooting		
Customer Care Executive		USN-8	I want to generate personalized energy-saving tips for customers during support interactions.	The system provides energy-saving recommendations tailored to the customer's specific usage patterns and needs	High	Srikantan

5 CODING & SOLUTIONING

5.1 Feature 1 – Time Series LSTM Prediction

Data Preprocessing:

Prepare the time series data, ensuring it is in the appropriate format for LSTM input.

Split the data into training and testing sets, maintaining the temporal order of the observations.

LSTM Model Training:

Define the architecture of the LSTM model, including the number of layers and units.

Configure the model with appropriate activation functions, loss functions, and optimizers.

Train the LSTM model using the training data, iterating through multiple epochs to optimize the model's performance.

Monitor the training process, evaluating the model's performance on validation data and making necessary adjustments, such as early stopping if the model starts to overfit.

Extract LSTM Features:

After training the LSTM model, utilize the model's internal representation or hidden states as features.

Pass the training and testing data through the LSTM layers and capture the output at a certain layer or time step.

Extract the LSTM features by considering the activations or states from the LSTM layers as inputs to subsequent models or analysis.

Feature Integration:

Combine the LSTM features with other relevant features from the dataset, such as weather data or occupancy information.

Perform any necessary feature engineering or transformations, such as normalization or scaling, to ensure consistency and meaningfulness of the features.

Model Integration and Prediction:

Incorporate the LSTM features into a larger prediction model, such as a regression model or another machine learning algorithm.

Train the integrated model using the combined features and the target variable (e.g., global active power).

Validate the model's performance using the testing data, assessing its ability to accurately predict the target variable based on the LSTM features and other input variables.

Evaluation and Iteration:

- Evaluate the performance of the LSTM-based feature approach by comparing the predicted values with the actual values in the testing data.
- Measure relevant evaluation metrics, such as root mean square error (RMSE), mean absolute error (MAE), or R-squared, to assess the accuracy and reliability of the predictions.
- Iterate on the feature selection, model architecture, or hyperparameter tuning process to further improve the accuracy and robustness of the predictions if necessary.

By utilizing LSTM as a feature in time series prediction, it becomes possible to capture the temporal dependencies and patterns in the data, enabling more

accurate and informative predictions for power consumption analysis in residential buildings.

5.2 Feature 2 – Chat Bot

We are planning to add a chatbot using OpenAI API which is used to answer questions regarding this power consumption project

Purpose of the Chatbot:

The chatbot serves as an interactive interface for users to ask questions and seek information related to electricity and power-related queries.

It enhances the user experience by providing immediate responses and addressing common inquiries, saving users time and effort in finding information.

Functionality of the Chatbot:

Natural Language Processing: The chatbot API utilizes advanced natural language processing techniques to understand user queries and generate appropriate responses.

Contextual Understanding: The chatbot can maintain context during conversations, allowing users to ask follow-up questions or provide additional details for more accurate responses.

Query Resolution: The chatbot API is designed to answer a wide range of electricity and power-related queries, such as energy-saving tips, appliance recommendations, or explanations of energy concepts.

Error Handling: The chatbot can gracefully handle queries it cannot answer or ones that require further clarification. It can provide suggestions, direct users to relevant resources, or escalate the query to a human agent if necessary.

Integration with Power Consumption Analysis:

Access to Project Data: The chatbot can access relevant data from the power consumption analysis, such as predicted energy consumption patterns, insights, and recommendations.

Query-Specific Responses: The chatbot can provide personalized responses based on the user's energy usage data or specific context, allowing for tailored recommendations or insights.

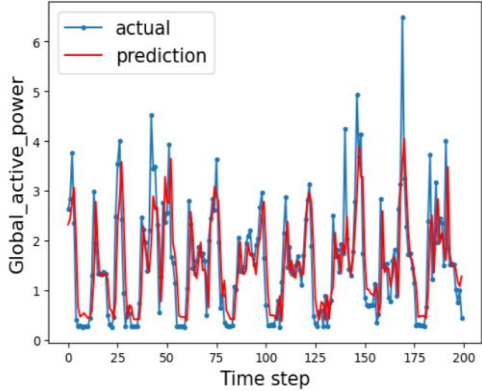
Integration with ML Models: The chatbot can leverage the trained ML models to provide accurate predictions or forecasts when users inquire about future energy consumption or trends.

6 RESULTS

6.1 Performance Metrics

Model Performance Testing:

Project team shall fill the following information in the model performance testing template.

S.No	Parameter	Values	Screenshot
1.	Metrics	Regression – Linear Regression	MAE: 0.028166653709573594 MSE: 0.0019182811907734647 RMSE: 0.04379818707176662 RSquarevalue: 0.9984900811880753
		Time Series Regression Model: LSTM For time series analysis	
2	Other models trained	Random Forest model Evaluation	MAE: 0.02125336923687959 MSE: 0.0013550051990774547 RMSE: 0.036810395258370356 RSquarevalue: 0.9989334473745645

		Ridge Forest model Evaluation	MAE: 0.02816670284006896 MSE: 0.001918282201835233 RMSE: 0.043798198614043854 RSquarevalue: 0.9984900803922476
		# XGBRegressor model Evaluation	MAE: 0.020945526682806907 MSE: 0.001209749856950687 RMSE: 0.03478145852247555 RSquarevalue: 0.9990477808594909

7 ADVANTAGES & DISADVANTAGES

Advantages of the power consumption analysis for households ML project:

1. **Energy conservation:** By accurately predicting energy consumption patterns, homeowners and utility companies can identify opportunities to reduce waste, optimize energy usage, and promote energy conservation. This can lead to significant cost savings and a reduced environmental footprint.
2. **Cost savings:** With insights into energy usage patterns, homeowners can make informed decisions about when and how to use energy-intensive appliances, taking advantage of off-peak hours or lower electricity tariffs. This can result in cost savings on utility bills.
3. **Sustainability:** The project contributes to sustainability efforts by promoting efficient energy management and reducing carbon emissions. By optimizing energy consumption, homeowners can minimize their impact on the environment and support a greener future.
4. **Data-driven decision-making:** The use of machine learning models allows for data-driven decision-making. Homeowners and utility companies can rely on accurate predictions and analysis to make informed choices regarding energy usage, equipment upgrades, and sustainability initiatives.
5. **Increased awareness:** The project raises awareness about energy consumption patterns and encourages individuals to adopt energy-saving practices. Through visualizations and recommendations, homeowners can better understand their energy usage habits and take steps to reduce waste.

Disadvantages and challenges of the power consumption analysis for households ML project:

1. **Data quality and availability:** The accuracy and availability of the collected data can impact the effectiveness of the ML models. Inaccurate or incomplete

data may lead to less reliable predictions and insights. Ensuring data quality and establishing data collection processes can be challenging.

2. Complexity of energy usage: Energy consumption is influenced by various factors, including weather conditions, occupant behavior, and individual preferences. Capturing and incorporating all these variables accurately in the analysis can be challenging, making it difficult to create comprehensive models.
3. Lack of real-time data: Most energy consumption data is collected at intervals (e.g., minute-averaged), which may not provide real-time insights. Real-time data availability could further enhance the project's effectiveness in dynamically responding to changes in energy demand and supply.
4. Limited user engagement: While the project may provide accurate predictions and recommendations, user engagement and adoption of suggested changes can vary. Convincing homeowners to actively participate and implement energy-saving measures may be a challenge, limiting the project's impact.
5. Regulatory and infrastructure constraints: Implementation of certain energy-saving measures may be subject to regulatory constraints or infrastructure limitations. For example, installing smart meters or integrating with existing utility infrastructure may require significant investment and regulatory approvals.

Addressing these challenges requires continuous improvement and adaptation of the project, including data quality control measures, user education and engagement strategies, and collaboration with stakeholders such as utility companies and regulatory bodies. Despite these challenges, the advantages of the power consumption analysis for households ML project can lead to significant benefits in energy management, cost savings, and sustainability.

8 CONCLUSION

In conclusion, the power consumption analysis for households ML project has provided valuable insights into energy usage patterns and has the potential to make a significant impact on sustainability. By utilizing machine learning algorithms, accurate predictions of future energy consumption can be made, enabling homeowners and utility companies to better manage their energy usage, reduce waste, and lower costs. The project's dataset, including parameters such as global active power, global reactive power, voltage, and sub-metering, has proven to be instrumental in training the models and identifying energy usage patterns. The inclusion of an inbuilt chatbot further enhances the project's usability by addressing queries and providing guidance on electricity and power-related issues. Overall, this project serves as a valuable tool in promoting efficient energy management, contributing to a greener and more sustainable future.

9 FUTURE SCOPE

Some of the future works which can be done from this project are:

1. **Weather integration:** Incorporate weather data such as temperature, humidity, and weather conditions into the analysis. By analyzing the correlation between weather patterns and energy consumption, you can identify how external factors influence power usage. For example, high temperatures may lead to increased air conditioning usage, while colder weather can result in higher heating demands.
2. **Demand forecasting:** Use historical electricity demand data, along with other relevant factors such as time of day, day of the week, and seasonal variations, to build models for predicting electricity demand in the future. This can help homeowners and utility companies anticipate peak demand periods and make informed decisions on energy usage and distribution.
3. **Supply analysis:** Analyze the supply side of electricity by considering factors like grid load, power generation sources, and renewable energy availability. This analysis can help identify periods of high or low supply, facilitating better management of electricity resources and promoting the utilization of renewable energy sources when they are most abundant.
4. **Demand response programs:** Incorporate demand response strategies into the analysis, which involve modifying electricity usage patterns based on supply-demand dynamics. By encouraging homeowners to shift their energy consumption to off-peak hours or reduce usage during times of high demand, the project can help balance the grid and optimize energy distribution.
5. **Market prices and tariffs:** Consider integrating market price data and electricity tariffs into the analysis. By factoring in the cost of electricity at different times of the day or during peak hours, homeowners can make informed decisions on when to use energy-intensive appliances or engage in energy-saving practices.
6. **Dynamic pricing and incentives:** Develop models that suggest dynamic pricing strategies or provide incentives to homeowners based on predicted electricity demand and supply conditions. This can encourage users to adjust their energy consumption patterns to align with periods of lower demand or higher availability of renewable energy, ultimately reducing overall energy costs and promoting sustainability.

10 APPENDIX

Source Codes:

index.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Energy Consumption Analysis</title>
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css') }}">
</head>
<body style="background-image: linear-gradient(to bottom, hsl(200, 100%, 90%), hsl(240,
100%, 90%));">
  <header>
    <div class="title">
      <h1>Energy Consumption Analysis System</h1>
    </div>
    <div class="index-buttons">
      <button><a href="{{ url_for('index') }}">Home</a></button>
      <button><a href="{{ url_for('inspect') }}">Inspect</a></button>
    </div>
  </header>
  <div class="container">
    <div class="description-container">
      <h1 style="font-family: 'Sacramento', cursive;">Welcome to our project on analyzing
energy consumption of energy-efficient appliances.</h1>
      <p>Click <a href="{{ url_for('inspect') }}">here</a> to inspect and analyze the
energy consumption.</p>
    </div>
  </div>
</body>
</html>
```

inspect.html

```
<!DOCTYPE html>
<html style="background-image: linear-gradient(to bottom, hsl(120, 50%, 90%), hsl(120,
50%, 70%));">
<head>
  <title>Energy Consumption Analysis</title>
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css') }}">
```



```

</head>
<body>
  <header>
    <div class="title">
      <h1>Energy Consumption Analysis System</h1>
    </div>
    <div class="index-buttons">
      <button><a href="{{ url_for('index') }}">Home</a></button>
      <button><a href="{{ url_for('inspect') }}">Inspect</a></button>
    </div>
  </header>
  <div class="container">
    <h1>Energy Consumption Inspection</h1>
    <form action="{{ url_for('inspect') }}" method="POST">
      <label for="GlobalReactivePower">Global Reactive Power:</label>
      <input type="text" name="GlobalReactivePower" required><br><br>

      <label for="Global_intensity">Global Intensity:</label>
      <input type="text" name="Global_intensity" required><br><br>

      <label for="Sub_metering_1">Sub-metering 1:</label>
      <input type="text" name="Sub_metering_1" required><br><br>

      <label for="Sub_metering_2">Sub-metering 2:</label>
      <input type="text" name="Sub_metering_2" required><br><br>

      <label for="Sub_metering_3">Sub-metering 3:</label>
      <input type="text" name="Sub_metering_3" required><br><br>

      <input type="submit" value="Submit">
    </form>
  </div>
</body>
</html>

```

output.html

```

<!DOCTYPE html>
<html style="background-image: linear-gradient(to bottom, hsl(30, 100%, 90%), hsl(60, 100%, 90%));">
<head>
  <title>Energy Consumption Analysis</title>
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>

```

```

<header>
  <div class="title">
    <h1>Energy Consumption Analysis System</h1>
  </div>
  <div class="index-buttons">
    <button><a href="{{ url_for('index') }}">Home</a></button>
    <button><a href="{{ url_for('inspect') }}">Inspect</a></button>
  </div>
</header>
<div class="container">
  <div class="result-container">
    <h1>Energy Consumption Prediction Result</h1>
    <p>The predicted global active power is: {{ output1 }} Watt</p>
    <h1>Useful suggestions and carbon foot print</h1>
    <p>{{ output2 }}</p>
  </div>
</div>
</body>
</html>

```

style.css

```
/* Common Styles */
```

```
body {
  background-image: url("../bg_image.jpg");
  margin: 0;
  padding: 0;
  font-family: Arial, sans-serif;
}
```

```
/* Index Page Styles */
```

```
header {
  background-color: #333;
  padding: 20px;
  display: flex;
  justify-content: space-between;
  align-items: center;
}
.title {
  flex: 1;
}
.title h1 {
  font-size: 24px;
  color: #fff;
  margin: 0;
}
```

```

.index-buttons {
  flex: 1;
  text-align: right;
}
.index-buttons button {
  background-color: #fff;
  border: none;
  color: #333;
  padding: 10px 20px;
  margin-left: 10px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.index-buttons button:hover {
  background-color: #333;
  color: #fff;
}
.container {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}
.image-container {
  flex: 1;
}
.image-container img {
  max-width: 100%;
  height: auto;
}
.description-container {
  flex: 1;
  padding-left: 20px;
}
.description-container p {
  font-size: 18px;
  line-height: 1.5;
}
.description-container a {
  color: #333;
  font-weight: bold;
  text-decoration: none;
  transition: color 0.3s ease;
}

```

```

}
.description-container a:hover {
    color: #ff5500;
}
/* Inspect Page Styles */
form {
    max-width: 400px;
    margin: 0 auto;
}
form label {
    font-size: 18px;
    margin-bottom: 10px;
}

form input {
    width: 100%;
    padding: 10px;
    margin-bottom: 20px;
    font-size: 16px;
}
form input[type="submit"] {
    width: 100%;
    padding: 10px;
    font-size: 18px;
    background-color: #333;
    color: #fff;
    border: none;
    cursor: pointer;
    transition: background-color 0.3s ease;
}
form input[type="submit"]:hover {
    background-color: #ff5500;
}
/* Output Page Styles */
.result-container {
    text-align: center;
    padding: 50px;
}
.result-container h1 {
    font-size: 24px;
    color: #333;
    margin-bottom: 20px;
}
.result-container p {

```

```

    font-size: 18px;
    color: #555;
}
html {
    background-image: url("NM Project/bg_image.jpg");
    background-repeat: no-repeat;
    background-size: cover;
}

```

```

nm_notebook.py
import openai

```

```

API_KEY = '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'
openai.api_key = API_KEY
model_id = 'gpt-3.5-turbo'

```

```

def ChatGPT_conversation(conversation):
    response = openai.ChatCompletion.create(
        model=model_id,
        messages=conversation
    )
    conversation.append({'role': response.choices[0].message.role, 'content':
response.choices[0].message.content})
    return conversation

```

```

conversation = []
prompt = 'imagine you are an expert in energy efficiency, my global active power is 1000
watts give predictions. assume average is 10 watts. give carbon foot print,i am in india,just
give me a number,dont tell as an ai model stuff'
conversation.append({'role': 'user', 'content': prompt})
conversation = ChatGPT_conversation(conversation)
q = conversation[-1]['content'].strip()

```

```

from flask import Flask, request, render_template
import pickle

```

```

model = pickle.load(open('PCASSS_model.pkl', 'rb'))
app = Flask(__name__)

```

```

@app.route("/")
def index():
    return render_template("index.html")

```

```

@app.route("/inspect", methods=["GET", "POST"])

```

```

def inspect():
    if request.method == "POST":
        GlobalReactivePower = float(request.form['GlobalReactivePower'])
        Global_intensity = float(request.form['Global_intensity'])
        Sub_metering_1 = float(request.form['Sub_metering_1'])
        Sub_metering_2 = float(request.form['Sub_metering_2'])
        Sub_metering_3 = float(request.form['Sub_metering_3'])
        x = [[GlobalReactivePower, Global_intensity, Sub_metering_1, Sub_metering_2,
Sub_metering_3]]
        output = str(round(model.predict(x)[0], 3))
        conversation = []
        prompt = 'imagine you are a expert in energy efficiency, my global active power is
'+output+ 'watts give predictions. assume average is 10watts. give carbon foot print,i am in
india,just give me a number,dont tell as an ai model stuff,dont tell statements like cant give a
number etc'
        conversation.append({'role': 'user', 'content': prompt})
        conversation = ChatGPT_conversation(conversation)
        q = conversation[-1]['content'].strip()
        return render_template('output.html', output1=output,output2 = q)
    return render_template("inspect.html")

```

```

if __name__ == "__main__":
    app.run(debug=True)

```

```

NM_Project_Notebook.ipynb
from google.colab import drive
drive.mount('/content/drive')

```

```

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import r2_score
from sklearn.linear_model import LinearRegression
from xgboost import XGBRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import Ridge

```

```

dataset = pd.read_csv("/content/drive/MyDrive/Household_power_consumption.csv")

```

```

dataset.head()
dataset.tail()
dataset.info()

```

```

dataset.isnull().sum()
dataset.loc[dataset.Sub_metering_3.isnull()].head()

dataset.replace('?', np.nan, inplace=True)
dataset.loc[dataset.Sub_metering_3.isnull()].head()

dataset = dataset.dropna()
for i in dataset.columns[2:] :
    dataset[i] = dataset[i].astype('float64')

dataset.shape
dataset.isnull().sum().sum()

values = dataset.values
dataset['Sub_metering_4'] = (values[:,2] * 1000 / 60) - (values[:,6] + values[:,7] + values[:,8])
dataset['Sub_metering_4'] = dataset['Sub_metering_4'].astype('float64')

dataset.dtypes
dataset

dataset.describe()
dataset.corr()

sns.distplot(dataset['Global_active_power'])
sns.distplot(dataset['Global_reactive_power'], kde=False, bins=30)
sns.distplot(dataset['Global_active_power'], kde=False, bins=30)

sns.jointplot( x='Global_reactive_power', y='Global_active_power', data=dataset,
kind='scatter' )
sns.jointplot( x='Voltage', y='Global_active_power', data=dataset, kind='scatter' )
sns.jointplot( x='Global_intensity', y='Global_active_power', data=dataset, kind='scatter' )
sns.jointplot( x='Sub_metering_1', y='Global_active_power', data=dataset, kind='scatter' )
sns.jointplot( x='Sub_metering_3', y='Global_active_power', data=dataset, kind='scatter' )
sns.jointplot( x='Sub_metering_4', y='Global_active_power', data=dataset, kind='scatter' )

pearson = dataset.corr(method='pearson')
mask = np.zeros_like(pearson)
mask[np.triu_indices_from(mask)] = True
sns.heatmap(pearson, vmax=1, vmin=0, square=True, cbar=True, annot=True,
cmap='YlGnBu', mask=mask)

X = dataset.iloc[:,[3,5,6,7,8]]
y = dataset.iloc[:,2]

```

```
X.head()
```

```
y.head()
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
lm.fit(X_train, y_train)
predictions = lm.predict(X_test)
predictions
```

```
from xgboost import XGBRegressor
model2=XGBRegressor()
model2.fit(X_train,y_train)
y_predict2 = model2.predict(X_test)
y_predict2
```

```
from sklearn.ensemble import RandomForestRegressor
model1 = RandomForestRegressor()
model1.fit(X_train,y_train)
y_predict1 = model1.predict(X_test)
y_predict1
```

```
from sklearn.linear_model import Ridge
model3 = Ridge()
model3.fit(X_train,y_train)
y_predict3 = model3.predict(X_test)
y_predict3
```

```
y_p1 = lm.predict([[0.418,18.4,0.0,1.0,17.0]])
y_p1
```

```
## Linear Regression model Evaluation
from sklearn import metrics
print('MAE:',metrics.mean_absolute_error(y_test,predictions))
print('MSE:',metrics.mean_squared_error(y_test, predictions))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test,predictions)))
print('RSquarevalue:',metrics.r2_score(y_test,predictions))
```



```
# XGBRegressor model Evaluation
from sklearn import metrics
from sklearn import metrics
print('MAE:',metrics.mean_absolute_error(y_test,y_predict2))
print('MSE:',metrics.mean_squared_error(y_test, y_predict2))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test,y_predict2)))
print('RSquarevalue:',metrics.r2_score(y_test,y_predict2))
```

```
# Random Forest model Evaluation
from sklearn import metrics
from sklearn import metrics
print('MAE:',metrics.mean_absolute_error(y_test,y_predict1))
print('MSE:',metrics.mean_squared_error(y_test, y_predict1))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test,y_predict1)))
print('RSquarevalue:',metrics.r2_score(y_test,y_predict1))
```

```
# Ridge Forest model Evaluation
from sklearn import metrics
from sklearn import metrics
print('MAE:',metrics.mean_absolute_error(y_test,y_predict3))
print('MSE:',metrics.mean_squared_error(y_test, y_predict3))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test,y_predict3)))
print('RSquarevalue:',metrics.r2_score(y_test,y_predict3))
```

```
from sklearn.model_selection import cross_val_score
cv = cross_val_score(lm,X,y,cv=5)
np.mean(cv)
```

```
Time_series_data_analysis_using_LSTM.ipynb
from google.colab import drive
drive.mount('/content/drive')
```

```
# Let`s import all packages that we may need :
import sys
import numpy as np # linear algebra
from scipy.stats import randint
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv), data manipulation as
in SQL
import matplotlib.pyplot as plt # this is used for the plot the graph
import seaborn as sns # used for plot interactive graph.
from sklearn.model_selection import train_test_split # to split the data into two parts
from sklearn.model_selection import KFold # use for cross validation
from sklearn.preprocessing import StandardScaler # for normalization
from sklearn.preprocessing import MinMaxScaler
```

```

from sklearn.pipeline import Pipeline # pipeline making
from sklearn.model_selection import cross_val_score
from sklearn.feature_selection import SelectFromModel
from sklearn import metrics # for the check the error and accuracy of the model
from sklearn.metrics import mean_squared_error, r2_score

## for Deep-learning :
import keras
from keras.layers import Dense
from keras.models import Sequential
from keras.utils import to_categorical
from keras.optimizers import SGD
from keras.callbacks import EarlyStopping
from keras.utils import np_utils
import itertools
from keras.layers import LSTM
from keras.layers.convolutional import Conv1D
from keras.layers.convolutional import MaxPooling1D
from keras.layers import Dropout

## Data can be downloaded from: http://archive.ics.uci.edu/ml/machine-learning-databases/00235/
## Just open the zip file and grab the file 'household_power_consumption.txt' put it in the
directory
## that you would like to run the code.
df = pd.read_csv('/content/drive/MyDrive/household_power_consumption.txt', sep=';',
                parse_dates={'dt': ['Date', 'Time']}, infer_datetime_format=True,
                low_memory=False, na_values=['nan', '?'], index_col='dt')

df.head()
df.info()
df.dtypes
df.shape
df.describe()
df.columns

## finding all columns that have nan:
dropping_list_all=[]
for j in range(0,7):
    if not df.iloc[:, j].notnull().all():
        dropping_list_all.append(j)
        #print(df.iloc[:,j].unique())
dropping_list_all

```

```

# filling nan with mean in any columns

for j in range(0,7):
    df.iloc[:,j]=df.iloc[:,j].fillna(df.iloc[:,j].mean())
# another sanity check to make sure that there are not more any nan
df.isnull().sum()

def series_to_supervised(data, n_in=1, n_out=1, dropnan=True):
    n_vars = 1 if type(data) is list else data.shape[1]
    dff = pd.DataFrame(data)
    cols, names = list(), list()
    # input sequence (t-n, ... t-1)
    for i in range(n_in, 0, -1):
        cols.append(dff.shift(i))
        names += [('var%d(t-%d)' % (j+1, i)) for j in range(n_vars)]
    # forecast sequence (t, t+1, ... t+n)
    for i in range(0, n_out):
        cols.append(dff.shift(-i))
        if i == 0:
            names += [('var%d(t)' % (j+1)) for j in range(n_vars)]
        else:
            names += [('var%d(t+%d)' % (j+1, i)) for j in range(n_vars)]
    # put it all together
    agg = pd.concat(cols, axis=1)
    agg.columns = names
    # drop rows with NaN values
    if dropnan:
        agg.dropna(inplace=True)
    return agg

## resampling of data over hour
df_resample = df.resample('h').mean()
df_resample.shape

values = df_resample.values

# normalize features
scaler = MinMaxScaler(feature_range=(0, 1))
scaled = scaler.fit_transform(values)
# frame as supervised learning
reframed = series_to_supervised(scaled, 1, 1)

# drop columns we don't want to predict
reframed.drop(reframed.columns[[8,9,10,11,12,13]], axis=1, inplace=True)

```

```

print(reframed.head())

# split into train and test sets
values = reframed.values

n_train_time = 365*24
train = values[:n_train_time, :]
test = values[n_train_time:, :]
##test = values[n_train_time:n_test_time, :]
# split into input and outputs
train_X, train_y = train[:, :-1], train[:, -1]
test_X, test_y = test[:, :-1], test[:, -1]
# reshape input to be 3D [samples, timesteps, features]
train_X = train_X.reshape((train_X.shape[0], 1, train_X.shape[1]))
test_X = test_X.reshape((test_X.shape[0], 1, test_X.shape[1]))
print(train_X.shape, train_y.shape, test_X.shape, test_y.shape)
# We reshaped the input into the 3D format as expected by LSTMs, namely [samples,
timesteps, features].

model = Sequential()
model.add(LSTM(100, input_shape=(train_X.shape[1], train_X.shape[2])))
model.add(Dropout(0.2))
# model.add(LSTM(70))
# model.add(Dropout(0.3))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')

# fit network
history = model.fit(train_X, train_y, epochs=20, batch_size=70, validation_data=(test_X,
test_y), verbose=2, shuffle=False)

# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper right')
plt.show()

# make a prediction
yhat = model.predict(test_X)
test_X = test_X.reshape((test_X.shape[0], 7))
# invert scaling for forecast

```

```

inv_yhat = np.concatenate((yhat, test_X[:, -6:]), axis=1)
inv_yhat = scaler.inverse_transform(inv_yhat)
inv_yhat = inv_yhat[:,0]
# invert scaling for actual
test_y = test_y.reshape((len(test_y), 1))
inv_y = np.concatenate((test_y, test_X[:, -6:]), axis=1)
inv_y = scaler.inverse_transform(inv_y)
inv_y = inv_y[:,0]
# calculate RMSE
rmse = np.sqrt(mean_squared_error(inv_y, inv_yhat))
print("Test RMSE: %.3f" % rmse)

## time steps, every step is one hour (you can easily convert the time step to the actual time
index)
## for a demonstration purpose, I only compare the predictions in 200 hours.

aa=[x for x in range(200)]
plt.plot(aa, inv_y[:200], marker='.', label="actual")
plt.plot(aa, inv_yhat[:200], 'r', label="prediction")
plt.ylabel('Global_active_power', size=15)
plt.xlabel('Time step', size=15)
plt.legend(fontsize=15)
plt.show()

```

GITHUB LINK:

<https://github.com/naanmudhalvan-SI/PBL-NT-GP--7992-1681102296>