NAAN MUTHALVAN

IBM COLLABARATE

ARTIFICIAL INTELLIGENCE

PROJECT TITLE

MEASURE ENERGY CONSUMPTION

NAME : LOGESH E
DEPT & YEAR : CSE & III yr
REG.NO : 712221104004
COLLEGE : PARK COLLEGE OF ENGINEERING AND TECHNOLOGY

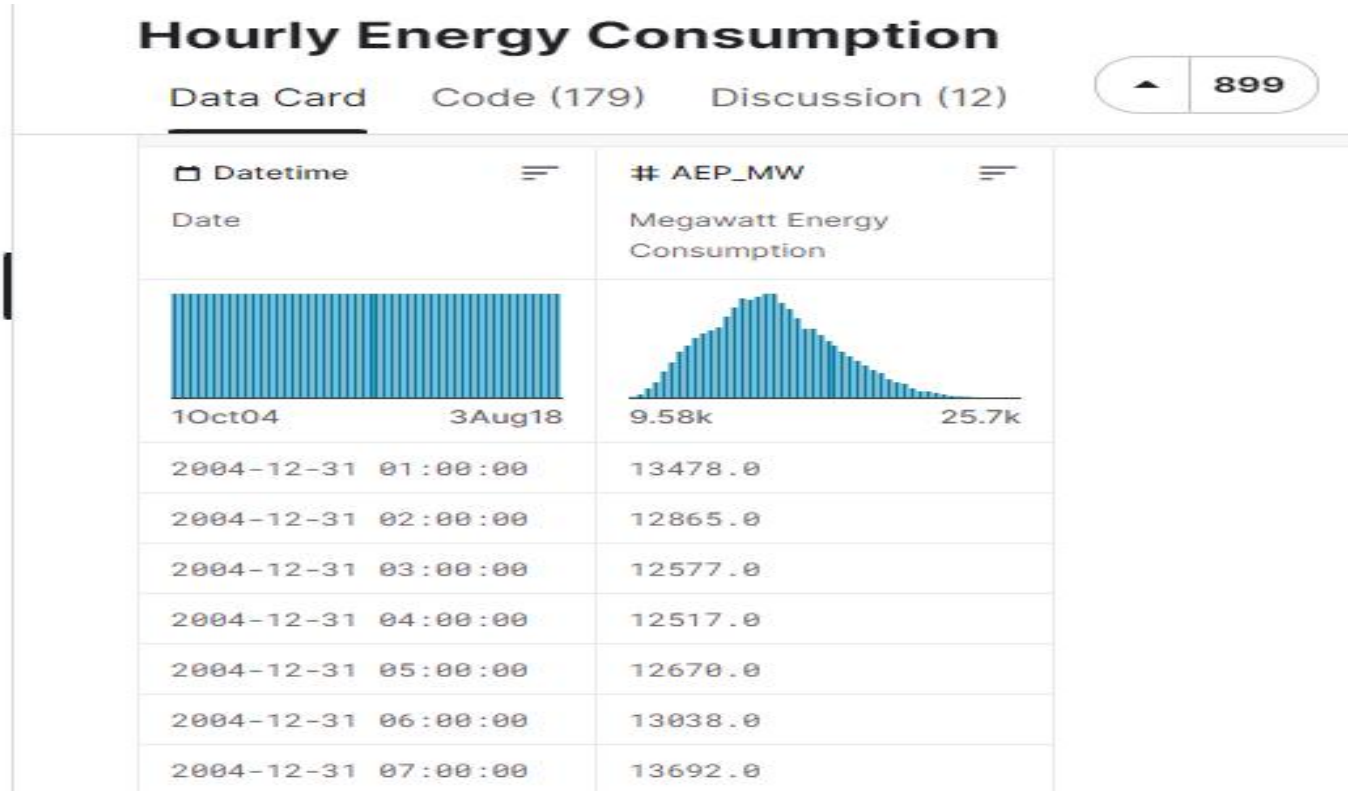
PROJECT INNOVATION IDEA

Steps :

1. Data Collection
2. Data Preprocessing
3. Build Machine Learning Model
4. Create Real Time Monitoring System
5. Build User Engagement
6. Data Analytics And Data Visualization
7. Deployment the Project

1. Data Collection

- ❖ Collect the data from various sources using smart meter and sensors.
- ❖ And also collect the weather data,occupancy information and other sources.
- ❖ Ensure data privacy and security compliance.



Hourly Energy Consumption

Data Card | Code (179) | Discussion (12) | 899

Datetime Date	AEP_MW Megawatt Energy Consumption
1Oct04 – 3Aug18	9.58k – 25.7k
2004-12-31 01:00:00	13478.0
2004-12-31 02:00:00	12865.0
2004-12-31 03:00:00	12577.0
2004-12-31 04:00:00	12517.0
2004-12-31 05:00:00	12670.0
2004-12-31 06:00:00	13038.0
2004-12-31 07:00:00	13692.0

Link : https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption

This link provide the data source collection about energy consumption.

2.Data Preprocessing

1. Data Cleaning

Why Data cleaning is needed because we are collecting from various sources the data might in unordered,anomaly and many format data are presented.so the data cleaning process to clean these data for required format.

This data cleaning mainly focused in three purpose

1. Missing data handling
2. Outlier detection
3. Duplicated data
4.

Technology needed : Python language are mostly used and library such as pandas.

2. Data Transformation

Hence the cleaning data are transform into categorical data .In data there are many categories like text,photo,numerical ,etc.. so we need to transform this data for particular format.

This process contain 4 division

1. Feature engineering
2. Normalization and Scaling
3. Encoding categorical data
4. Text processing

Technology needed : Python libraries like NumPy and Scikit-Learn are often used for numerical transformations, while Pandas is useful for data manipulation. Text processing can be handled with libraries like NLTK or spaCy.

3. Data Integration

In data integration is used for Combine data from multiple sources

Technology : SQL or Python libraries like Pandas can be used for merging and joining datasets.

4.. Data Reduction:

Reduce the dimensionality of your dataset to speed up processing and reduce noise. Techniques include Principal Component Analysis (PCA) for numerical data and feature selection for machine learning models.

Technology: Python libraries like Scikit-Learn for dimensionality reduction techniques.

5. Data Splitting:

Divide the dataset into training, validation, and test sets. This is essential for evaluating machine learning models properly.

Technology: Scikit-Learn or other machine learning libraries in Python.

6. Data Scaling and Transformation:

Apply scaling and transformations to the training and test datasets separately to avoid data leakage from the test set to the training set.

Technology: Scikit-Learn for scaling and transformation.

7. Data Imbalance Handling:

If your dataset has imbalanced classes, apply techniques like oversampling, undersampling, or using appropriate evaluation metrics to address the issue.

Technology: Python libraries like imbalanced-learn or custom code.

8. Data Validation:

Continuously validate the data throughout the project to ensure that it remains consistent and accurate.

Technology: Custom scripts and validation routines.

Notebook Input Output

In [1]:

```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load in

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list the files in the input directory

from subprocess import check_output
print(check_output(["ls", "../input"]).decode("utf8"))
```

```
---------------------------------------------------------------------------
CalledProcessError                        Traceback (most recent call last)
<ipython-input-1-fde84f249a2a> in <module>()
     10
     11 from subprocess import check_output
---> 12 print(check_output(["ls", "../input"]).decode("utf8"))
     13
     14 # Any results you write to the current directory are saved as output.

/opt/conda/lib/python3.6/subprocess.py in check_output(timeout, *popenargs, **kwargs)
    334
```

```
    335     return run(*popenargs, stdout=PIPE, timeout=timeou
t, check=True,
--> 336                **kwargs).stdout
    337
    338

/opt/conda/lib/python3.6/subprocess.py in run(input, timeout,
check, *popenargs, **kwargs)
    416         if check and retcode:
    417             raise CalledProcessError(retcode, process.
args,
--> 418                                      output=stdout, st
derr=stderr)
    419     return CompletedProcess(process.args, retcode, std
out, stderr)
```

```
/opt/conda/lib/python3.6/subprocess.py in run(input, timeout,
check, *popenargs, **kwargs)
    416         if check and retcode:
    417             raise CalledProcessError(retcode, process.
args,
--> 418                                      output=stdout, st
derr=stderr)
    419     return CompletedProcess(process.args, retcode, std
out, stderr)
    420

CalledProcessError: Command '['ls', '../input']' returned non-
zero exit status 2.
```

3.Build Machine Learning Model

In machine learning model this is the main part of the of project.Because this model understand the entire data process and act to that data future purpose.

Hence so many machine learning algorithm are used for predicting the future events.

Here we use 13 machine learning algorithms

1. Linear Regression:

Linear regression is suitable for predicting energy consumption as a function of various factors (e.g., time of day, temperature, occupancy). It provides a linear relationship between input variables and energy usage.

2. Time Series Analysis:

Time series analysis, including methods like ARIMA (AutoRegressive Integrated Moving Average), can be useful for modeling and forecasting energy consumption over time.

3. Neural Networks:

Neural networks, including feedforward networks and recurrent neural networks (RNNs), can capture complex patterns in energy data. Long Short-Term Memory (LSTM) networks are especially effective for modeling sequential energy consumption data.

4. Decision Trees and Random Forest:

Decision trees and random forests can be employed for regression tasks, providing interpretable models for energy consumption prediction.

5. Support Vector Machines (SVM):

SVMs are effective for both classification and regression tasks and can be used to model energy consumption patterns based on historical data.

6. Gradient Boosting:

Gradient boosting algorithms like XGBoost, LightGBM, and CatBoost can handle complex, non-linear relationships in energy consumption data.

7. Clustering Algorithms:

Clustering algorithms such as k-means or hierarchical clustering can help identify patterns and segment energy consumption data into groups with similar behavior.

8. Anomaly Detection Algorithms:

Algorithms for anomaly detection, like Isolation Forest or One-Class SVM, can be applied to identify unusual spikes or drops in energy consumption that may indicate issues or inefficiencies.

9. Reinforcement Learning:

Reinforcement learning can be used to optimize energy consumption in real-time by learning and adapting to changing conditions.

10. Feature Engineering:

Feature engineering techniques can help extract meaningful features from the data, such as creating lag features, aggregating data over time intervals, and incorporating weather or occupancy information.

11. Ensemble Methods:

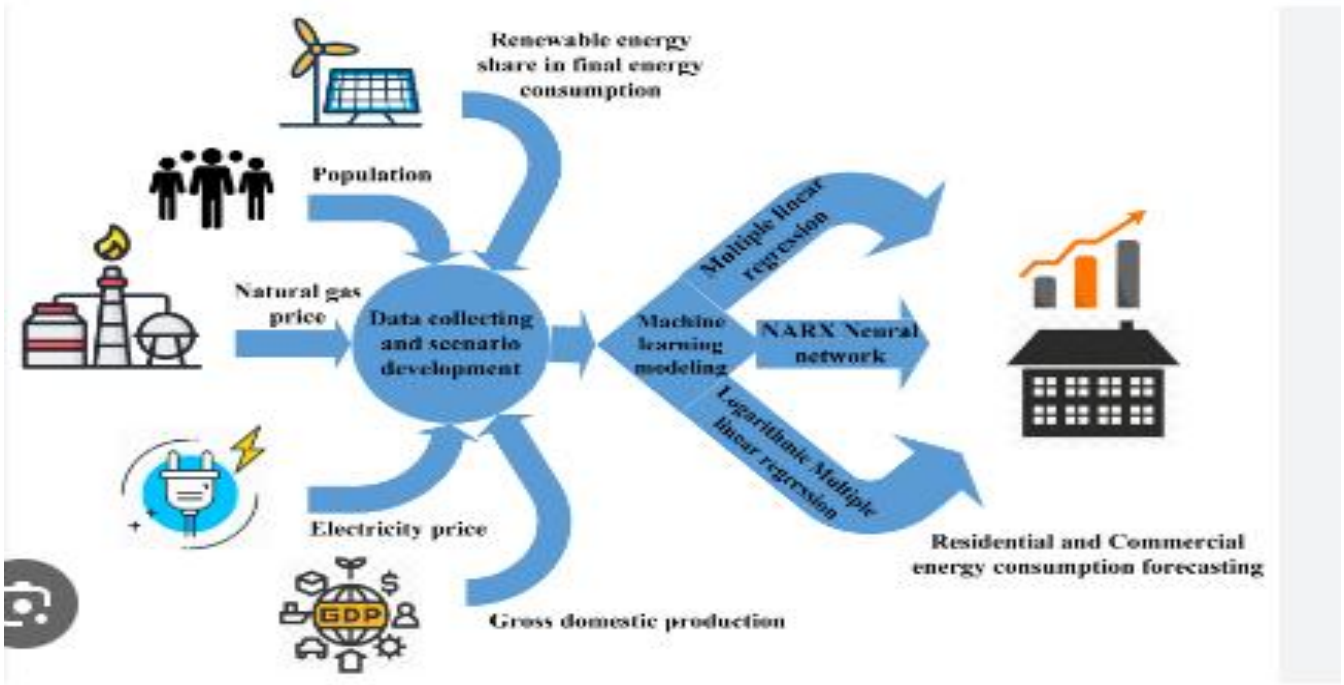
Ensemble methods like bagging and stacking can combine the predictions of multiple models to improve accuracy.

12. Deep Learning:

Deep learning models, such as deep neural networks and convolutional neural networks (CNNs), can be used for energy consumption prediction, especially in image-based analysis or complex scenarios.

13. AutoML:

AutoML (Automated Machine Learning) platforms can automatically select and tune machine learning algorithms for energy consumption prediction tasks.



4. Create Real Monitoring System

This monitoring system shows the all details about our energy consumption to user view.Many monitoring systems are in application format so here we are creating monitor system in web application type.

Hence we show the real monitoring system needed

1. Data Sources:

Define the data sources that will feed information to the dashboard. This can include sensor data, APIs, databases, logs, or any other relevant data streams.

2. Data Ingestion:

Implement mechanisms to ingest and process data from various sources in real time. This may involve technologies like Kafka, RabbitMQ, or custom scripts to capture and queue incoming data.

3. Data Processing:

Process and clean incoming data. This step can involve filtering, transformation, and aggregation to ensure that the data is in a usable format.

4. Database or Storage:

Store processed data in a suitable database or storage system. For real-time purposes, NoSQL databases like MongoDB, Cassandra, or time-series databases like InfluxDB are often used.

5. Backend and API:

Create a backend server that exposes APIs for retrieving real-time data. This server should handle user authentication, data retrieval, and updates.

6. Frontend Dashboard:

Build the user interface for the real-time dashboard using web technologies. This dashboard should have a user-friendly and intuitive design, and it may include features such as:
Real-time data updates: Use technologies like WebSockets, Server-Sent Events (SSE), or GraphQL subscriptions to enable real-time updates without the need for constant polling.
Interactive data visualization: Use JavaScript libraries like D3.js, Chart.js, or Plotly for creating dynamic charts and graphs.
Alerts and notifications: Implement notifications or alarms that trigger when specific thresholds are crossed.
User customization: Allow users to customize the dashboard layout, select data sources, or set their preferences.

7. Data Display:

Display the real-time data in various formats, including tables, graphs, maps, and widgets that are relevant to the monitored parameters.

8. User Authentication and Authorization:

Implement user authentication and authorization to ensure that only authorized individuals can access the dashboard and specific data.

9. Data Security:

Ensure that the data, both in transit and at rest, is secure. Use encryption, access control, and other security measures.

10. Mobile Responsiveness: -

Design the dashboard to be mobile-responsive, allowing users to access and monitor data from mobile devices.

11.Scalability:

Design the dashboard to handle increasing loads as the number of users and data sources grows. Consider load balancing and distributed architectures.

12.Technology Stack:

Front end : HTML,CSS,JAVA SCRIPT (like angular)
Real time updates : Websockets
Database : MongoDN,influxDB
Libraries: D3.js, Chart.js, Plotly for data visualization.

13.Hosting and Deployment:

Cloud Platforms : AWS,Azure,Google
Deployment Dashboard : Docker,kubernetics

5. User Engagement

This user engagement system provides the capability to use all the user in this system.

Hence the above process are used for user engage to use this system.

1. Personalization:

Personalization tailors content and recommendations to individual user preferences, making the experience more relevant and engaging.

Technologies: Collaborative filtering, content-based filtering, and recommendation algorithms.

2. Gamification:

Gamification techniques, such as points, badges, leaderboards, and challenges, can motivate users to take specific actions.

Technologies: Gamification platforms and frameworks for adding game elements to your application.

3. Push Notifications:

Send timely and relevant push notifications to users to keep them informed and engaged.

Technologies: Mobile app push notification services and web push notification platforms.

4. In-App Messaging:

In-app messaging allows you to communicate directly with users within your app, providing updates, tips, or support.

Technologies: In-app messaging SDKs and APIs.

5. Social Sharing and Community Building:

Encourage users to share content and participate in discussions or communities related to your product or service.

Technologies: Social media integration, forums, and chat platforms.

6. A/B Testing:

A/B testing helps you optimize your user interface, content, and features by experimenting with different variations and measuring user responses.

Technologies: A/B testing platforms and tools.

7. Content Quality:

High-quality, informative, and engaging content is essential for retaining and attracting users.

Technologies: Content management systems, content creation tools, and SEO optimization.

8. User-Friendly Design:

Intuitive and user-friendly design enhances user experience and encourages user engagement.

Technologies: UI/UX design tools and frameworks.

9. User Feedback and Surveys:

Collect feedback from users to understand their needs and preferences. Use surveys, feedback forms, and user analytics.

Technologies: Survey and feedback tools, user analytics platforms.

10. Analytical Tools: -

Analytical tools provide insights into user behavior and help you identify areas where engagement can be improved. -

Technologies: Google Analytics, Mixpanel, Kissmetrics, and custom analytics solutions.

11. Chatbots and Virtual Assistants:

Chatbots and virtual assistants can provide instant support, answer questions, and engage users in interactive conversations.

Technologies: Chatbot development platforms, NLP (Natural Language Processing) libraries, and AI chatbot frameworks.

12. Mobile Optimization:

Optimize your application for mobile devices to ensure a smooth and engaging user experience.

Technologies: Mobile app development frameworks, responsive design, and mobile app testing tools.

13. Feedback Loop:

Establish a feedback loop with users by acknowledging their input and showing them that their opinions matter.

14. Community Building:

Encourage user-generated content and community participation. Support forums, comment sections, and user-generated reviews can foster engagement.

15.Continuous Improvement:

Continuously gather and analyze data to understand user behavior, and use this information to make iterative improvements to your product or service.

16.Technology Stack:

web and mobile app development frameworks, database systems, analytics tools, content management systems, and customer relationship management (CRM) platforms.

6. Data Analytics And Data Visualization

This part shows the data analytics and visualization details. If any error present in the project this will show the where the error could be created.

Data Analysis Tools:

Python: Libraries like Pandas, NumPy, and Scikit-Learn.

R: A programming language specifically designed for statistical analysis.

Jupyter Notebooks: An interactive environment for data analysis and visualization.

SQL: Essential for querying and extracting data from databases.

Data Visualization:

Tableau: A powerful data visualization tool with a user-friendly interface.

Power BI: Microsoft's business intelligence platform for creating interactive reports and dashboards.
QlikView and Qlik Sense: Tools for creating interactive data visualizations.

D3.js: A JavaScript library for creating custom and interactive visualizations.

Reporting and Dashboard Tools:

Microsoft Excel: A popular tool for creating reports and dashboards.

Google Data Studio: A free tool for creating interactive reports and dashboards.

Apache Superset: An open-source data exploration and visualization platform.

Looker: A data exploration and business intelligence platform.
Predictive Analytics and Machine Learning:

Python: Libraries like Scikit-Learn, TensorFlow, and PyTorch for building machine learning models.

7. Deployment The Project

Here is the last process of our project because we are going to deployment our project to our environment.
Then check the all process if the systems is running properly It will show good else the system has some problem.

These above all process contain the information about how to measure energy consumption using AI.