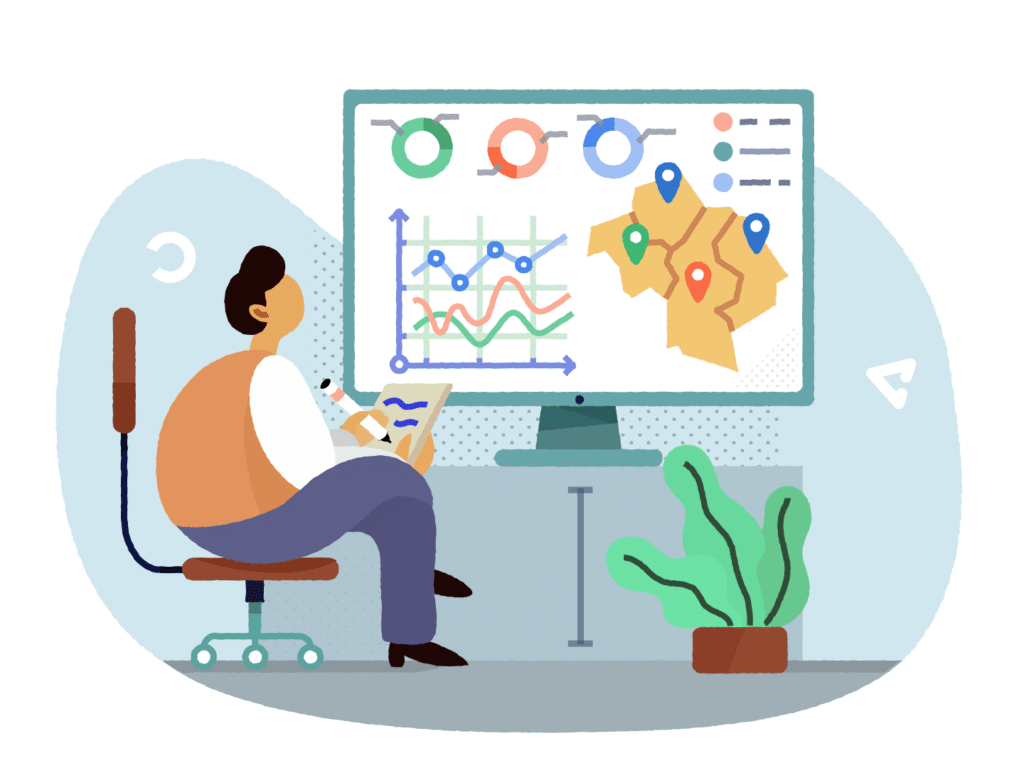
**Measure energy consumption**

**Team member Harish.N(au210221104010)**

**Phase 3:** **Development Part 1**

**Topic:** **Start building the model to measure energy consumption by loading and pre-processing the dataset.**

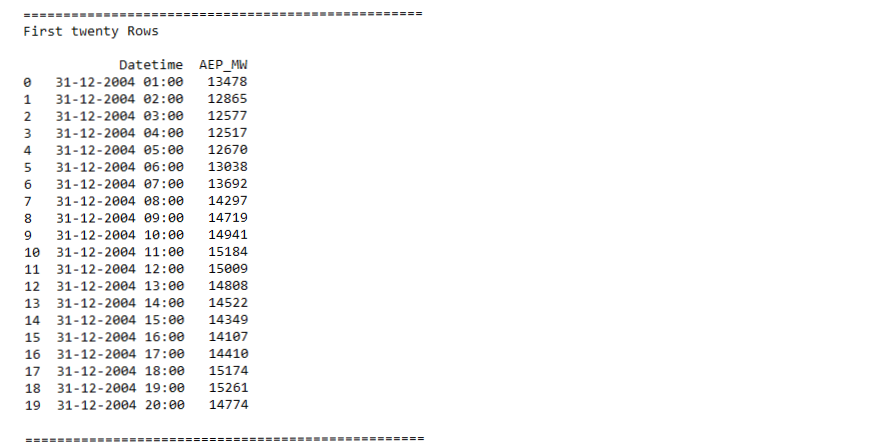
**Introduction**

* In the quest for a sustainable and efficient energy future, machine learning has emerged as a powerful tool for measuring and optimizing energy consumption. The journey towards harnessing the potential of machine learning begins with dataset loading and preprocessing, which forms the critical groundwork for accurate predictions and actionable insights.
* This introductory phase involves gathering diverse data sources, refining their quality, and shaping them into a format that machine learning models can comprehend. By effectively mastering the art of dataset loading and preprocessing, we unlock the door to enhanced energy efficiency, demand forecasting, and informed decision-making in our increasingly energy-conscious world.

**Given dataset**

(AEP\_hourly)

**121273x2(121273 rows and 2 column)**

****

**Importance of loading and processing dataset:**

**Data Quality Assurance**: Loading and preprocessing a dataset for measuring energy consumption using machine learning is crucial for ensuring data quality. This involves identifying and addressing missing values, handling outliers, and transforming data into a consistent format. For example, Python code using Pandas can be employed to fill missing values with 'SimpleImputer' and identify outliers using statistical methods like Z-scores or IQR.

**Model Performance**: Proper dataset loading and preprocessing significantly impact the performance of machine learning models. Techniques such as feature scaling, encoding categorical variables, and feature engineering help models make accurate predictions. Python code using libraries like Scikit-Learn can standardize features using 'StandardScaler' and one-hot encode categorical data with 'get\_dummies'

**Efficient Analysis**: Loading and preprocessing the dataset prepares it for efficient analysis. By converting data into a format that machine learning algorithms can understand, it becomes possible to extract valuable insights, make predictions, and optimize energy consumption effectively. This efficient analysis contributes to better decision-making and more sustainable energy management.

**Challenges involved while loading and preprocessing dataset:**

* Loading and preprocessing datasets for measuring energy consumption using machine learning can be a complex task, and it comes with several challenges. Some common challenges include:
* **Data Quality**: Ensuring the data's quality is a significant challenge. This includes dealing with missing values, errors, and inconsistencies in the dataset, which can lead to incorrect modeling results.
* **Outlier Detection**: Identifying and handling outliers is crucial in energy consumption data. Incorrect handling of outliers can lead to models with poor generalization.
* **Feature Engineering**: Creating meaningful features from raw data requires domain knowledge and can be time-consuming. Deciding which features to engineer and how to engineer them effectively is a challenge.

**How to overcome the challenges while loading and preprocessing dataset:**

* Overcoming challenges in loading and preprocessing datasets for measuring energy consumption using machine learning requires a combination of best practices, domain knowledge, and the use of appropriate tools. Here are some strategies to tackle these challenges:
* **Data Quality**:
* Imputation: Use techniques like mean, median, or mode imputation to address missing data.
* Data Validation: Implement data validation checks to detect and correct errors and inconsistencies in the dataset.
* **Outlier Detection**:
* Use statistical methods or machine learning algorithms to identify and handle outliers appropriately. Techniques like z-scores, isolation forests, or local outlier factor (LOF) can be useful.
* **Feature Engineering**:
* Collaborate with domain experts to create meaningful features that capture important relationships and patterns.
* Utilize automated feature selection and dimensionality reduction techniques to streamline feature engineering.

**Loading dataset**

* Loading the dataset using machine learning is the process of bringingthe data into the machine learning environment so that it can be usedto train and evaluate a model.
* The specific steps involved in loading the dataset will vary dependingon the machine learning library or framework that is being used.However, there are some general steps that are common to mostmachine learning frameworks:
* **Identify the dataset:**
* The first step is to identify the dataset that you want to load. This dataset may be stored in a local file, in a database, or in cloud storageservice.
* **Load the dataset:**
* Once you have identified the dataset, you need to load it into the machine learning environment. This may involve using a built-infunction in the machine learning library, or it may involve writing yourown code.
* **Preprocess the dataset:**
* Once the dataset is loaded into the machine learning environment,you may need to preprocess it before you can start training and evaluating your model. This may involve cleaning the data, transforming the data into a suitable format, and splitting the data into training and

test sets.

Here, how to load a dataset using machine learning in Python

**Program:**

**Import packages**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

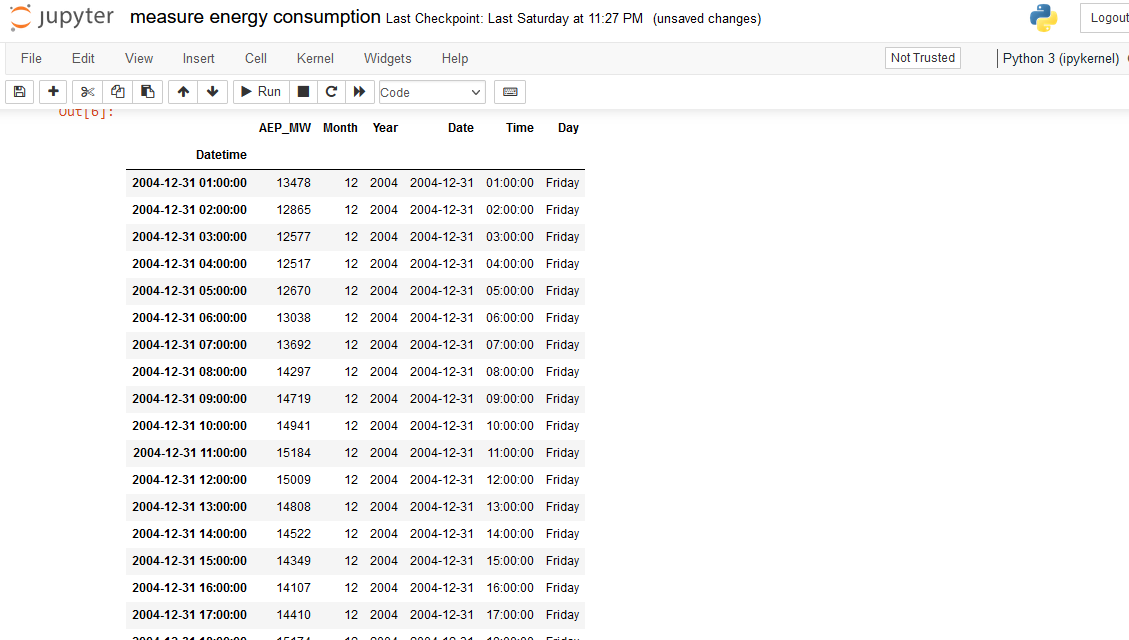
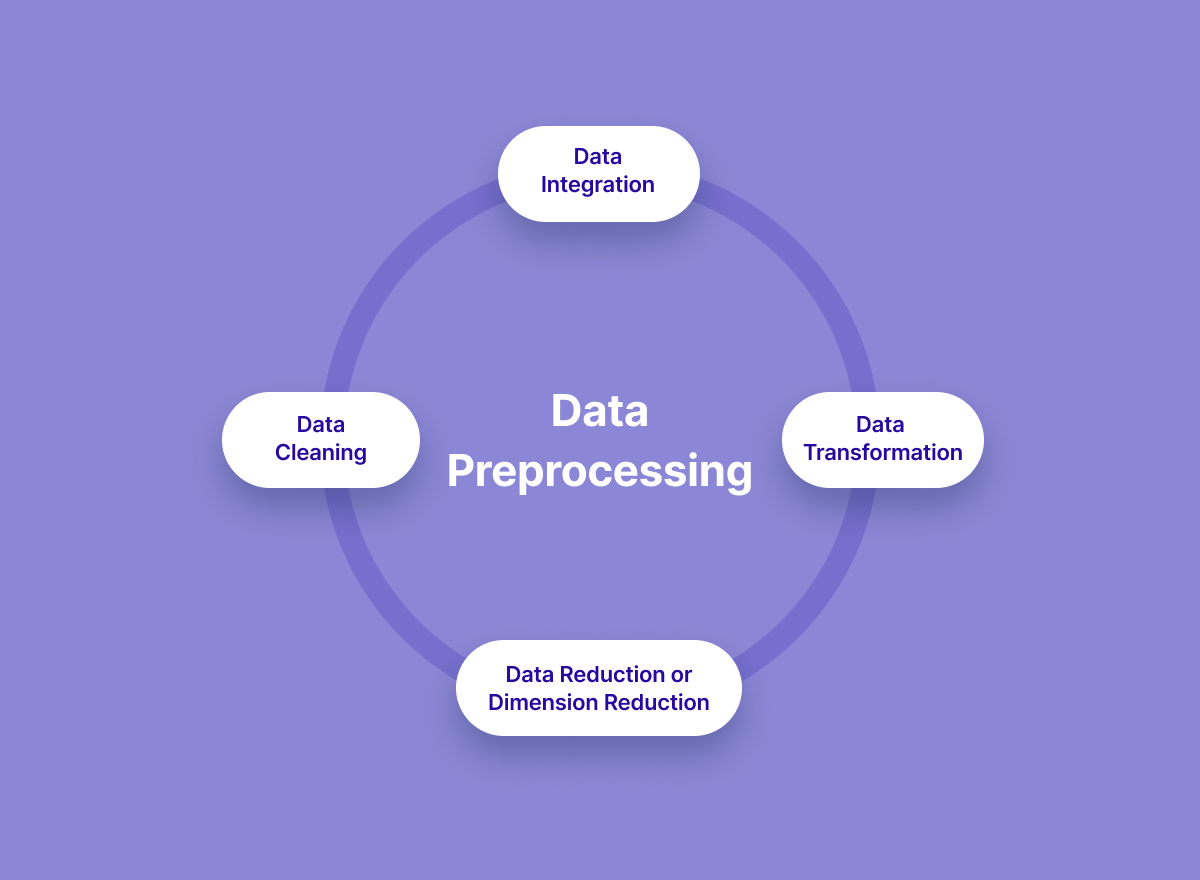
import pprint

%matplotlib inline

**Loading Dataset:**

df=pd.read\_csv(r"C:\Users\WELCOME\Documents\datasets\AEP\_hourly.csv")

**Reformat the Date Time Columns:**

Dataset: **Preprocessing the dataset**

* Data preprocessing is the process of cleaning, transforming, and integrating data in order to make it ready for analysis.
* This may involve removing errors and inconsistencies, handlingmissing values, transforming the data into a consistent format, and scaling the data to a suitable range.
* **Data cleaning**: This involves identifying and correcting errors and inconsistencies in the data. For example, this may involve removing duplicate records, correcting typos, and filling in missing values.
* **Data transformation**: This involves converting the data into a format that is suitable for the analysis task. For example, this may involve converting categorical data to numerical data, or scaling the data to a suitable range.
* **Feature engineering**: This involves creating new features from the existing data. For example, this may involve creating features that represent interactions between variables, or features that represent summary statistics of the data.
* **Data integration**: This involves combining data from multiple sources into a single dataset. This may involve resolving inconsistencies in the data, such as different data formats ordifferent variable names.
* Data preprocessing is an essential step in many data science projects. By carefully preprocessing the data, data scientists can improve the accuracy and reliability of their results

**Program**

**Import packages**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import pprint

%matplotlib inline

**Loading Dataset:**

df=pd.read\_csv(r"C:\Users\WELCOME\Documents\datasets\AEP\_hourly.csv")

**Exploratory Data Analysis:**

print("="\*50)

print("Information About Dataset","\n")

print(df.info(),"\n")

print("="\*50)

print("Describe the Dataset ","\n")

print(df.describe(),"\n")

print("="\*50)

print("Null Values t ","\n")

print(df.isnull().sum(),"\n")

**Resampleing Data**

NewDataSet = dataset.resample('D').mean()

print("Old Dataset ",dataset.shape )

print("New Dataset ",NewDataSet.shape )

TestData = NewDataSet.tail(100)

Training\_Set = NewDataSet.iloc[:,0:1]

Training\_Set = Training\_Set[:-60]

print("Training Set Shape ", Training\_Set.shape)

print("Test Set Shape ", TestData.shape)

Training\_Set = Training\_Set.values

sc = MinMaxScaler(feature\_range=(0, 1))

Train = sc.fit\_transform(Training\_Set)

X\_Train = []

Y\_Train = []

# Range should be fromm 60 Values to END

for i in range(60, Train.shape[0]):

# X\_Train 0-59

X\_Train.append(Train[i-60:i])

# Y Would be 60 th Value based on past 60 Values

Y\_Train.append(Train[i])

# Convert into Numpy Array

X\_Train = np.array(X\_Train)

Y\_Train = np.array(Y\_Train)

print(X\_Train.shape)

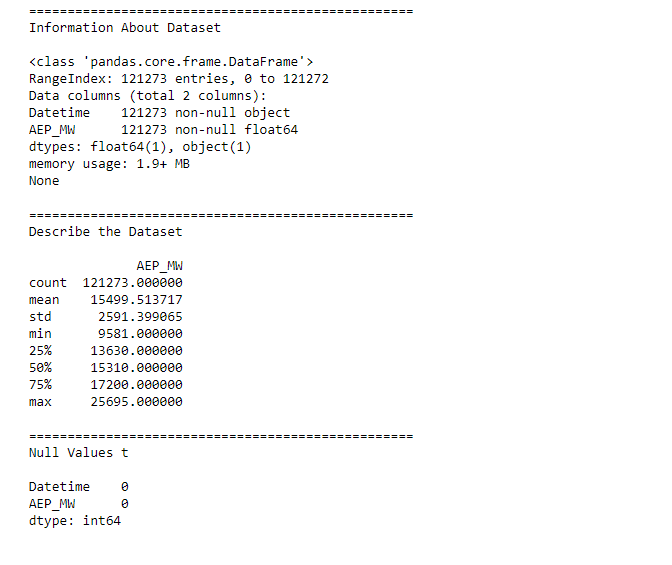
print(Y\_Train.shape)

# Shape should be Number of [Datapoints , Steps , 1 )

# we convert into 3-d Vector or #rd Dimesnsion

X\_Train = np.reshape(X\_Train, newshape=(X\_Train.shape[0], X\_Train.shape[1], 1))

X\_Train.shape

**Output Exploratory Data Analysis: **

**Conclusion**

In conclusion, dataset loading and preprocessing are the indispensable cornerstones of any machine learning endeavor aimed at measuring energy consumption. These initial steps, from data collection to data cleaning, feature engineering, and scaling, lay the foundation upon which accurate and insightful predictions are built. Careful data curation and meticulous preparation of the dataset are essential for ensuring the reliability and efficacy of machine learning models in the realm of energy consumption. By adhering to best practices in these processes, practitioners can harness the full potential of machine learning to optimize energy usage, predict demand, and make more informed decisions in the ever-evolving energy landscape.