**Measure Energy Consumption**

**Phase 5: Project Documentation & Submission**

Automated Energy Consumption Analysis System

The problem at hand is to create an automated system that measures energy consumption, analyses the data, and provides visualizations for informed decision-making. This solution aims to enhance efficiency, accuracy, and ease of understanding in managing energy consumption across various sectors.

Data Source

Identifying an Available Dataset To begin addressing this problem, the first step is to identify and acquire a suitable dataset containing energy consumption measurements. The dataset will serve as the foundation for our analysis and automation efforts.

Dataset

The code we provided appears to be reading a dataset from a CSV file named "AEP\_hourly.csv" using the Pandas library. It then performs data cleaning, analysis, and visualization on this dataset. The dataset seems to be related to hourly energy consumption, as evidenced by the column "AEP\_MW" (which likely represents megawatt usage) and the title of the histogram ("Histogram of MEGAWATT USAGE").

The specific details of the dataset, including its content, columns, and source, would be found in the "AEP\_hourly.csv" file, which is not included in the code snippet you provided. You would need access to this CSV file to understand the dataset's characteristics and content.

**Dataset ()**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import warnings

warnings.filterwarnings("ignore", category=**UserWarning**)

RED = "**\033**[91m"

GREEN = "**\033**[92m"

YELLOW = "**\033**[93m"

BLUE = "**\033**[94m"

RESET = "**\033**[0m"

df = pd.read\_csv("/kaggle/input/hourly-energy-consumption/AEP\_hourly.csv")

df["Datetime"] = pd.to\_datetime(df["Datetime"])

*# DATA CLEANING*

print(BLUE + "**\n**DATA CLEANING" + RESET)

*# --- Check for missing values*

missing\_values = df.isnull().sum()

print(GREEN + "Missing Values : " + RESET)

print(missing\_values)

*# --- Handle missing values*

df.dropna(inplace=True)

*# --- Check for duplicate values*

duplicate\_values = df.duplicated().sum()

print(GREEN + "Duplicate Values : " + RESET)

print(duplicate\_values)

*# --- Drop duplicate values*

df.drop\_duplicates(inplace=True)

*# DATA ANALYSIS*

print(BLUE + "**\n**DATA ANALYSIS" + RESET)

*# --- Summary Statistics*

summary\_stats = df.describe()

print(GREEN + "Summary Statistics : " + RESET)

print(summary\_stats)

*# DATA VISUALIZATION*

print(BLUE + "**\n**DATA VISUALIZATION" + RESET)

*# --- Line plot*

print(GREEN + "LinePlot : " + RESET)

plt.figure(figsize=(10, 6))

sns.lineplot(data=df, x="Datetime", y="AEP\_MW")

plt.xlabel("Datetime")

plt.ylabel("Energy Consumption (MW)")

plt.title("Energy Consumption Over Year")

plt.grid()

plt.show()

*# --- Histogram*

print(GREEN + "Histogram : " + RESET)

plt.figure(figsize=(10, 6))

plt.hist(

df["AEP\_MW"],

bins=100,

histtype="barstacked",

edgecolor="white",

)

plt.xlabel("AEPMW")

plt.ylabel("Frequency")

plt.title("Histogram of MEGAWATT USAGE")

plt.show()

This code is a Python script that does the following:

* It imports necessary libraries:
  + `pandas` for data manipulation and analysis.
  + `numpy` for numerical operations.
  + `matplotlib.pyplot` for creating data visualizations, such as plots and histograms.
  + `seaborn` for enhanced data visualization.
  + `warnings` to control how warning messages are displayed.
* 2. It sets some color codes for text formatting using ANSI escape sequences. These color codes are used to format printed messages in the terminal.
* 3. It reads a CSV file named "AEP\_hourly.csv" into a Pandas DataFrame named `df`. The dataset likely contains information about hourly energy consumption, and it includes a column named "Datetime."
* 4. It converts the "Datetime" column to datetime objects using `pd.to\_datetime`. This allows for easier manipulation of date and time information.
* 5. It performs data cleaning, analysis, and visualization in the following steps:
  + - \*\*Data Cleaning\*\*:
    - It checks for missing values in the DataFrame and prints the count of missing values.
    - It drops rows with missing values.
    - It checks for duplicate rows in the DataFrame and prints the count of duplicate rows.
    - It drops duplicate rows to ensure data consistency.
    - \*\*Data Analysis\*\*:
    - It computes and prints summary statistics for the DataFrame using `df.describe()`. This includes statistics like mean, standard deviation, and quartiles for numeric columns in the DataFrame.
    - \*\*Data Visualization\*\*:
    - It creates a line plot using Seaborn to visualize the "AEP\_MW" (megawatt usage) over time, with the x-axis representing datetime and the y-axis representing energy consumption in megawatts.
    - It creates a histogram to visualize the distribution of "AEP\_MW" values. The histogram is configured with 100 bins, bar stacking, and white edges.
* 6. Throughout the script, it uses colored text to print various messages in the terminal. For example, it uses `GREEN` and `BLUE` to indicate different sections of the code (e.g., data cleaning, data analysis, and data visualization).
* The code is essentially an example of a data analysis and visualization pipeline for a dataset related to hourly energy consumption, with an emphasis on cleaning, summarizing, and visualizing the data.

**Visualization**

DATA CLEANING

Missing Values :

Datetime 0

AEP\_MW 0

dtype: int64

Duplicate Values :

0

DATA ANALYSIS

Summary Statistics :

Datetime AEP\_MW

count 121273 121273.000000

mean 2011-09-02 03:17:01.553025024 15499.513717

min 2004-10-01 01:00:00 9581.000000

25% 2008-03-17 15:00:00 13630.000000

50% 2011-09-02 04:00:00 15310.000000

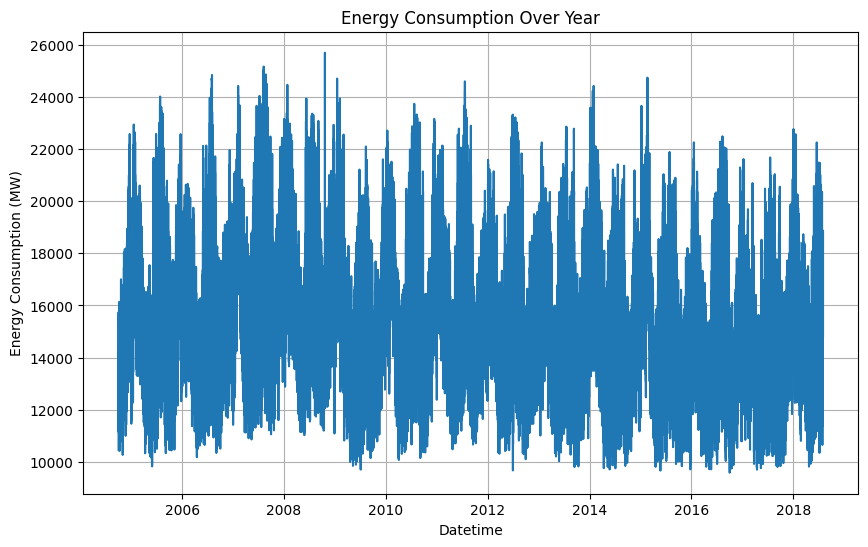
75% 2015-02-16 17:00:00 17200.000000

max 2018-08-03 00:00:00 25695.000000

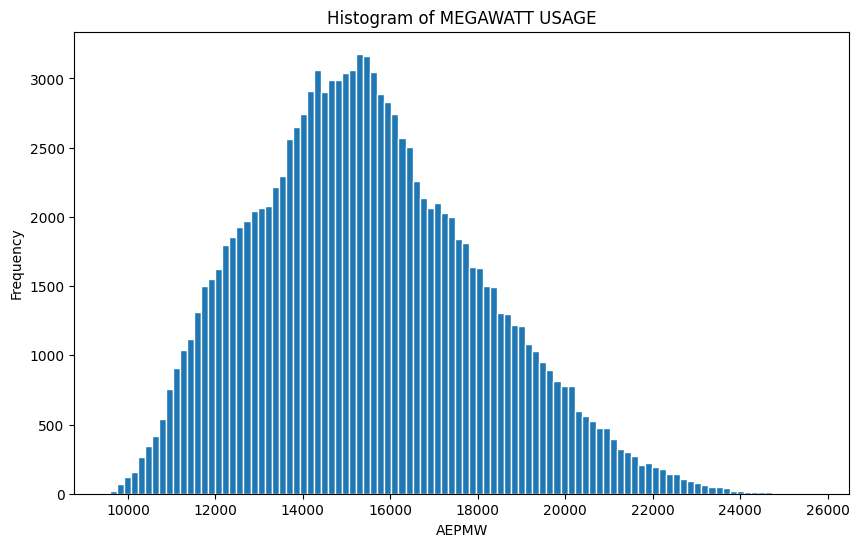
std NaN 2591.399065

DATA VISUALIZATION

LinePlot :



Histogram :



**INNOVATIVE**

Time series analysis is an innovative technique for analyzing and forecasting energy consumption data collected over time. It employs methods like ARIMA, LSTM, and Prophet to model and predict energy consumption patterns. This innovation is valuable as it enables better resource planning, cost reduction, and sustainability efforts. Energy providers can optimize resource allocation during peak demand, businesses and individuals can cut costs by aligning energy usage with forecasts, and organizations can reduce their carbon footprint. Moreover, time series analysis aids in energy market analysis, assisting traders and investors in making informed decisions. Ultimately, it empowers data-driven strategies, fostering efficient energy use and informed decision-making across the energy sector.

**Conclusion**

In conclusion, measuring and managing energy consumption is a critical endeavor with profound implications for sustainability, cost-efficiency, and environmental impact. The innovative techniques and approaches mentioned earlier, including time series analysis, predictive maintenance, geospatial analysis, and more, play a pivotal role in this domain.