**TEAM MEMBER**

**AU411521104032-K.ELAKIYA**

**Phase 2 Development part 1document**

PROJECT: MEASURE ENERGY CONSUMPTION



**INTRODUCTION**

* Measuring energy consumption in Python can be done in a number of ways, depending on the specific needs of the project.
* One common approach is to use a hardware sensor, such as a power meter or smart plug, to measure the power consumption of a device or appliance.
* This data can then be read and processed using Python to calculate the total energy consumption over a period of time.
* Another approach to measuring energy consumption in Python is to use a software tool, such as the Intel Running Average Power Limit (RAPL) interface.
* RAPL is a hardware feature that provides real-time power consumption measurements for Intel CPUs.
* Python libraries such as CodeCarbon can be used to access RAPL data and calculate the energy consumption of a Python program or script.
* Measure energy consumption is an important step in reducing energy waste and improving energy efficiency. Python can be used to measure energy consumption in a variety of ways, including:
* Using the Intel "Running Average Power Limit" (RAPL) technology to estimate the power consumption of a CPU. Using the pyJoules library to measure the energy consumption of a host machine along the execution of a piece of Python code.
* Once the energy consumption data has been collected, it can be pre-processed using Python to clean, transform, and prepare the data for analysis.
* This may involve handling missing values, converting data types, resampling the data, creating new features, removing outliers, and splitting the data into training and testing sets.
* Once the data is pre-processed, it can be used for machine learning or other analysis tasks. For example, machine learning models can be trained to predict energy consumption based on historical data and other factors.

**LOAD** **THE** **DATASET**

To load the dataset in measure energy consumption, we can use the following Python code:

import pandas as pd

# Load the dataset from a CSV file

energy\_consumption\_df = pd.read\_csv('energy\_consumption.csv')

This will load the dataset into a Pandas DataFrame, which is a convenient data structure for working with tabular data.

Once the dataset is loaded, we can explore it using various Pandas functions. For example, we can print the first few rows of the DataFrame to get a sense of the data:

Python

print(energy\_consumption\_df.head())

**EXPLORATORY DATA ANALYSIS**

Exploratory Data Analysis (EDA) is a process of investigating the data to discover patterns, trends, and relationships. EDA is an important step in any data science project, as it helps us to understand the data better and to identify potential problems.

Here are some steps involved in performing EDA on energy consumption data using Python:

* **Import the necessary libraries.**

Python

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

Use code with caution.

content\_copy

* **Load the dataset.**

Python

energy\_consumption\_df = pd.read\_csv('energy\_consumption.csv')

Use code with caution.

content\_copy

* **Check for missing values and outliers.**

Python

# Check for missing values

print(energy\_consumption\_df.isna().sum())

# Check for outliers

print(energy\_consumption\_df['energy\_consumption'].describe())

Use code with caution.

content\_copy

* **Explore the data.**

Python

# Print the first few rows of the DataFrame

print(energy\_consumption\_df.head())

# Get the data types of the columns

print(energy\_consumption\_df.dtypes)

# Calculate summary statistics for the dataset

print(energy\_consumption\_df.describe())

* **Visualize the data.**

Python

# Create a line plot of the energy consumption over time

plt.plot(energy\_consumption\_df['timestamp'], energy\_consumption\_df['energy\_consumption'])

plt.xlabel('Timestamp')

plt.ylabel('Energy consumption (Wh)')

plt.title('Energy consumption over time')

plt.show()

# Create a histogram of the energy consumption

sns.histplot(energy\_consumption\_df['energy\_consumption'])

plt.xlabel('Energy consumption (Wh)')

plt.ylabel('Frequency')

plt.title('Histogram of energy consumption')

plt.show()

# Create a box plot of the energy consumption by device type

sns.boxplot(x = 'device', y = 'energy\_consumption', data=energy\_consumption\_df)

plt.xlabel('Device type')

plt.ylabel('Energy consumption (Wh)')

plt.title('Box plot of energy consumption by device type')

plt.show()

content\_copy

* **Identify patterns and trends.**

Once we have explored the data, we can start to identify patterns and trends. For example, we may notice that the energy consumption is higher during certain times of day or during certain seasons. We may also notice that certain devices consume more energy than others.

* **Formulate hypotheses.**

Based on the patterns and trends that we have identified, we can start to formulate hypotheses about the energy consumption data. For example, we may hypothesize that the energy consumption is higher during the summer because people are using m``` ore air conditioning.

* **Test the hypotheses.**

Once we have formulated hypotheses, we can test them using statistical methods. For example, we can use a t-test to test the hypothesis that the energy consumption is higher during the summer.

* **Interpret the results.**

Once we have tested our hypotheses, we need to interpret the results. If a hypothesis is rejected, then we need to find a new explanation for the data. If a hypothesis is not rejected, then we can accept it as a valid explanation for the data.

EDA is an iterative process. We may need to repeat some of the steps above multiple times in order to fully understand the data.

Here are some additional things to keep in mind when performing EDA on energy consumption data:

* **Consider the time period of the data.** If the data is only for a short period of time, then it may not be representative of the overall energy consumption.
* **Consider the location of the data.** If the data is from a specific location, then it may not be representative of the energy consumption in other locations.
* **Consider the types of devices that are included in the data.** If the data only includes certain types of devices, then it may not be representative of the energy consumption of all devices.

Once we have performed EDA on the energy consumption data, we can use it to develop strategies for reducing energy consumption.

**FOR PHASE-3\_PROJECT :**

In this phase I’ve designed an innovation to solve the problem.

**DATA SOURCE :**

The data source for measuring the energy consumption is obtained from the below dataset.

DATASET:<https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption>

**MODULE :**

When it comes to measuring energy consumption, there are a few key modules that are typically involved:

**1️.Sensors:**

These devices are responsible for collecting data on energy usage. They can include electricity meters, smart plugs, or other monitoring devices.

**2️.Data Acquisition:**

This module focuses on gathering and recording the energy consumption data from the sensors. It ensures that the data is accurately captured and stored for further analysis.

**3️.Data Processing:**

This module involves analyzing and Processing the collected data. It often uses programming Languages like Python and data analysis tools like pandas to Calculate energy consumption metrics and identify patterns or Trends.

**4️.Visualization:**

This module helps to present the energy Consumption data in a visual format. It can include charts, graphs, Or dashboards that make it easier to understand and interpret the Data.These modules work together to provide insights into energy Usage and help identify opportunities for energy efficiency Improvements.

**MODEL DEVELOPMENT:**

To develop a model for measuring energy consumption, you can

Follow these steps:

**1️. Collect data:**

Gather energy consumption data from sensors or Smart meters. This data should include variables like time, date, and energy usage.

**2️. Pre-process the data:**

Clean the data by handling missing values, outliers, and formatting issues. You may need to convert he data into a suitable format for analysis.

**3️. Feature engineering:**

Extract relevant features from the data that can help in predicting energy consumption. This can include factors like weather conditions, occupancy, or time of day.

**4️. Split the data:**

Divide the dataset into training and testing sets. The training set is used to train the model, while the testing set is used to evaluate its performance.

**5. Choose and train a model:**

Select a suitable machine learning algorithm, such as linear regression, decision trees, or neural networks. Train the model using the training data.

**6. Evaluate the model:**

Assess the model’s performance using evaluation metrics like mean squared error or R-squared. This will help you understand how well the model predicts energy consumption.

**7. Fine-tune and validate:**

Adjust the model’s parameters and hyper parameters to improve its performance. Validate the model using cross-validation techniques to ensure its generalizability.

**8. Deploy and monitor:**

Once you’re satisfied with the model’s performance, deploy it in a production environment. Continuously monitor its predictions and update the model as needed.

**PYTHON PROGRAM FOR MEASURE ENERGY CONSUMPTION : EXAMPLE MODEL -1**

#python program for measuring energy consumption

Import pandas as pd

Import matplotlib.pyplot as plt

# Read energy consumption data from a CSV file

Data = pd.read\_csv(‘energy\_data.csv’)

To represent energy consumption using a bar diagram, you can

Use Python’s matplotlib library.

# Sample data

Categories = [‘Category 1️’, ‘Category 2️’, ‘Category 3️’]

Consumption = [1️0, 2️0, 1️5] # Energy consumption values

# Create the bar plot

Plt.bar(categories, consumption)

# Add labels and title

Plt.xlabel(‘Categories’)

Plt.ylabel(‘Energy Consumption’)

Plt.title(‘Energy Consumption by Category’)

# Show the plot

Plt.show()

This code creates a bar plot with categories on the x-axis and

Energy consumption values on the y-axis.

# Plot the energy consumption over time

Plt.plot(data[‘Date’], data[‘Energy Consumption’])

Plt.xlabel(‘Date’)

Plt.ylabel(‘Energy Consumption’)

Plt.title(‘Energy Consumption Over Time’)

Plt.show()

# plotting

Import matplotlib.pyplot as plt

Import numpy as np # linear algebra

Import os # accessing directory structure

Import pandas as pd # data processing, CSV file I/O (e.g.

Pd.read\_csv)

Import seaborn as sns

Plt.style.use(‘ggplot’) # Make it pretty

In [2️]:

# Data is saved in parquet format so schema is preserved.

Df = pd.read\_parquet(‘../input/est\_hourly.paruqet’)

**EXAMPLE MODEL – 2 :**

Import matplotlib.pyplot as plt # plotting

Import numpy as np # linear algebra

Import os # accessing directory structure

Import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)

Import seaborn as sns

Plt.style.use(‘ggplot’) # Make it pretty

In [2]:

# Data is saved in parquet format so schema is preserved.

Df = pd.read\_parquet(‘../input/est\_hourly.paruqet’)

Data index is the date/hour, columns are for different regions within PJM.

Regions joined at different times, so not all have data for all dates. Regions also split (PJM\_Load split to East and West)

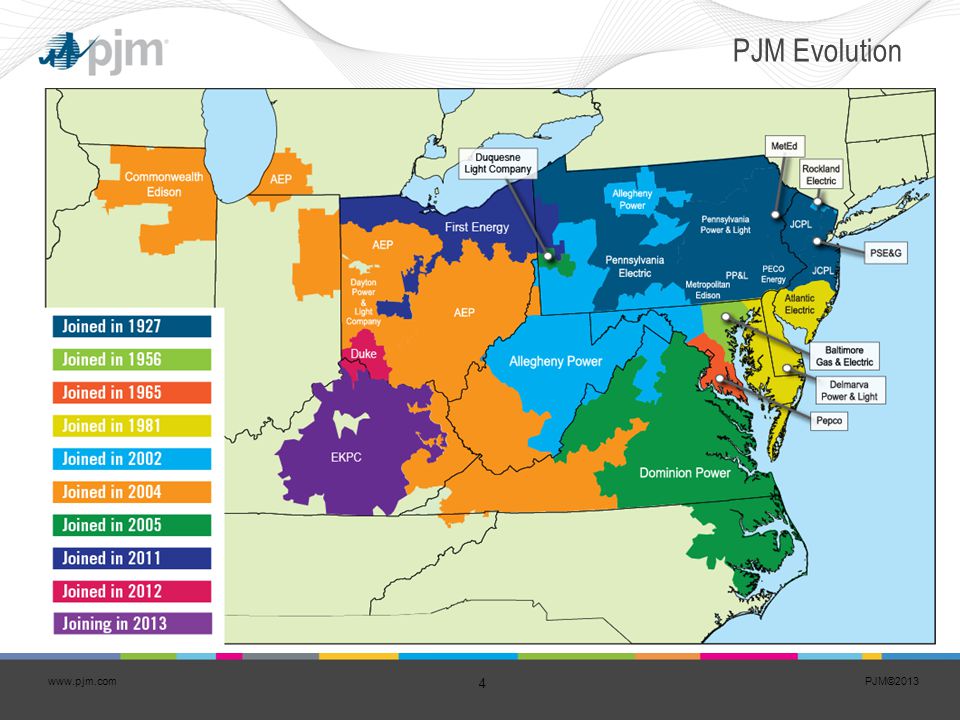
In [3]:

#Show PJM Regions

From IPython.display import Image

Image(url= <http://slideplayer.com/4238181/14/images/4/PJM+Evolution.jpg>)

Out[3]:

In [4]:

Df.head()

Out[4]:

In [5]:

**AEP COMED DAYTON DEOK DOM DUQ EKPC FE NI PJME PJMW PJM\_Load**

Datetime

1998-12-31 01:00:00 NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN 29309.0

1998-12-31 02:00:00 NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN 28236.0

1998-12-31 03:00:00 NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN 27692.0

1998-12-31 04:00:00 NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN 27596.0

1998-12-31 05:00:00 NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN 27888.0

Df.describe().T

Count mean std min 25% 50% 75% max

AEP 121273.0 15499.513717 2591.399065 9581.0 13630.0 15310.0 17200.00 25695.0

COMED 66497.0 11420.152112 2304.139517 7237.0 9780.0 11152.0 12510.00 23753.0

DAYTON 121275.0 2037.851140 393.403153 982.0 1749.0 2009.0 2279.00 3746.0

DEOK 57739.0 3105.096486 599.859026 907.0 2687.0 3013.0 3449.00 5445.0

DOM 116189.0 10949.203625 2413.946569 1253.0 9322.0 10501.0 12378.00 21651.0

DUQ 119068.0 1658.820296 301.740640 1014.0 1444.0 1630.0 1819.00 3054.0

EKPC 45334.0 1464.218423 378.868404 514.0 1185.0 1386.0 1699.00 3490.0

FE 62874.0 7792.159064 1331.268006 0.0 6807.0 7700.0 8556.00 14032.0

NI 58450.0 11701.682943 2371.498701 7003.0 9954.0 11521.0 12896.75 23631.0

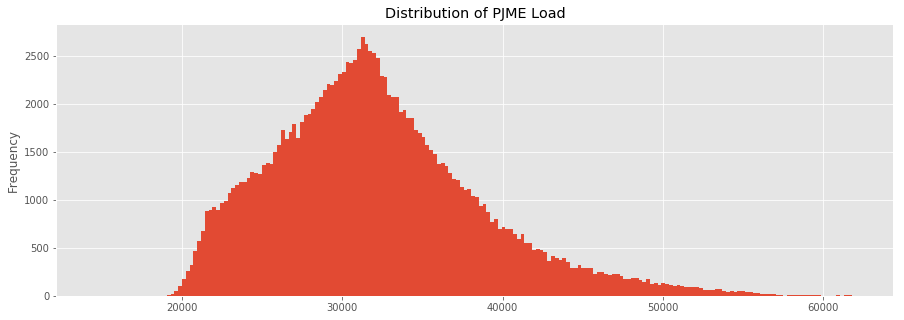
PJME 145366.0 32080.222831 6464.012166 14544.0 27573.0 31421.0 35650.00 62009.0

PJMW 143206.0 5602.375089 979.142872 487.0 4907.0 5530.0 6252.00 9594.0

PJM\_Load 32896.0 29766.427408 5849.769954 17461.0 25473.0 29655.0 33073.25 54030.0

\_ = df[‘PJME’].plot.hist(figsize=(15, 5), bins=200, title=’Distribution of PJME Load’)

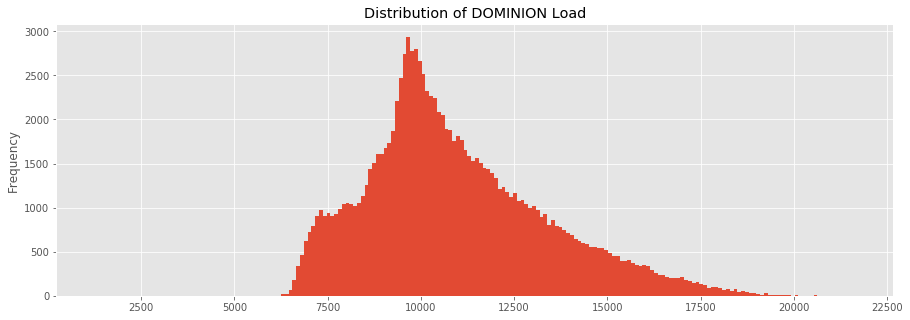
Df.describe().T\_ = df[‘PJME’].plot.hist(figsize=(15, 5), bins=200, title=’Distribution of PJME Load’)



Out[5]:

In [6]:

\_ = df[‘DOM’].plot.hist(figsize=(15, 5), bins=200, title=’Distribution of DOMINION Load’)



Import matplotlib.pyplot as plt # plotting

Import numpy as np # linear algebra

Import os # accessing directory structure

Import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)

Import seaborn as sns

Plt.style.use(‘ggplot’) # Make it pretty

# Data is saved in parquet format so schema is preserved.

Df = pd.read\_parquet(‘../input/est\_hourly.paruqet’)

Data index is the date/hour, columns are for different regions within PJM.

Create Time Series Features

In [13]:

Df[‘dow’] = df.index.dayofweek

Df[‘doy’] = df.index.dayofyear

Df[‘year’] = df.index.year

Df[‘month’] = df.index.month

Df[‘quarter’] = df.index.quarter

Df[‘hour’] = df.index.hour

Df[‘weekday’] = df.index.weekday\_name

Df[‘woy’] = df.index.weekofyear

Df[‘dom’] = df.index.day # Day of Month

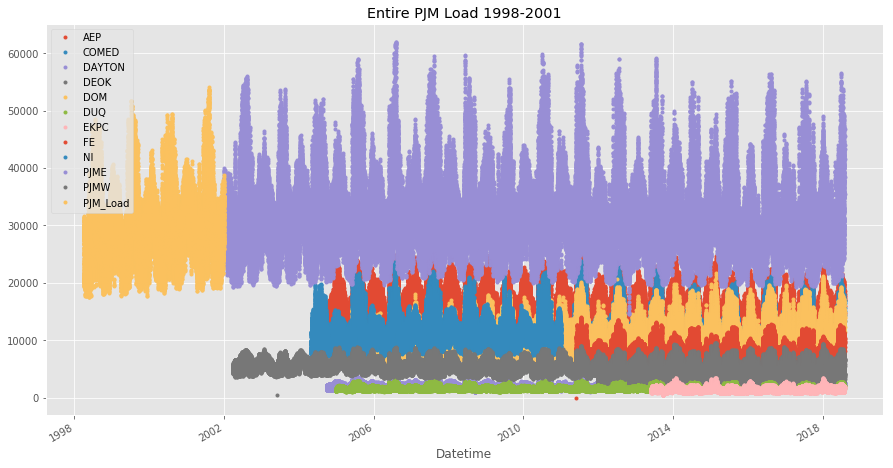
Df[‘date’] = df.index.date

In [14]:

## **Plot Time Series**

In [9]:

plot = df.plot(style='.', figsize=(15, 8), title='Entire PJM Load 1998-2001')

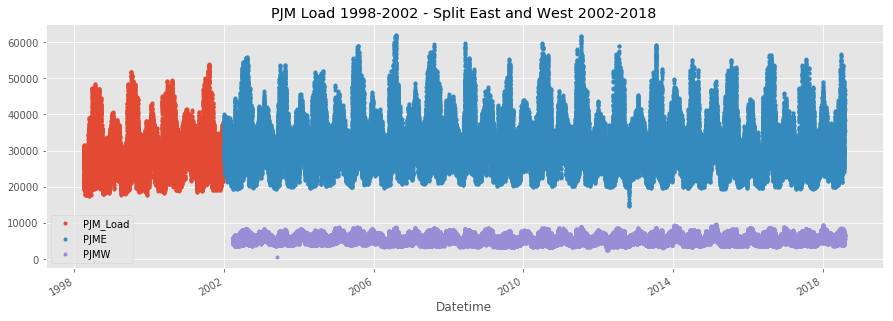


# **Plotting Regions**

In [10]:

\_ = df[['PJM\_Load','PJME','PJMW']] \

.plot(style='.', figsize=(15, 5), title='PJM Load 1998-2002 - Split East and West 2002-2018')



= df[[‘PJM\_Load’,’hour’]].plot(x=’hour’,

Y=’PJM\_Load’,

Kind=’scatter’,

Figsize=(14,4),

Title=’Consumption by Hour of Day’)

**CONCLUSION :**

So, when it comes to the conclusion for measuring energy consumption, it’s all about being aware of our energy usage and making conscious choices to reduce it. By measuring and monitoring our energy consumption, we can identify areas of improvement and implement energy-saving strategies. This not only helps us save money on our bills, but also contributes to a more sustainable and eco-friendly lifestyle. So let’s keep track of our energy usage and make a positive impact together.