AI Measure Energy Consumption

Name:Lobia.A

REG NO:720421104031

INTRODUCTION :

K-Nearest Neighbours (KNN) is a popular machine learning algorithm used for classification and regression tasks. It is a **lazy learning**, non-parametric algorithm that uses data with several classes to predict the classification of the new sample point. KNN is non-parametric since it doesn't make any assumptions on the data being studied.

During the training phase, the KNN algorithm stores the entire training dataset as a reference. When implementing an algorithm, you will always need a data set. So, you start by loading the training and the test data. Then, you choose the nearest data points (the value of K). K can be any integer.

The working of KNN Algorithm in Machine Learning can be summarized in three steps:

1. Load the data
2. Choose the nearest data points (the value of K)
3. Do the following, for each test data –
   - Calculate the distance between test data and each row of training data
   - Sort the calculated distances in ascending order based on distance values
   - Get top K rows from sorted array
   - Get the most frequent class of these rows
   - Return this class as output.

PROCESS:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

```python
import seaborn as sns


# customize the style
pd.options.display.float_format = '{:.5f}'.format
pd.options.display.max_rows = 12


# load the data
filepath = '../input/hourly-energy-consumption/PJME_hourly.csv'
df = pd.read_csv(filepath)


print("Now, you're ready for step one")


# turn data to datetime
df = df.set_index('Datetime')
df.index = pd.to_datetime(df.index)


# create the plot
df.plot(style='.',
        figsize=(15, 5),
        title='PJM Energy (in MW) over time')
plt.show()
```
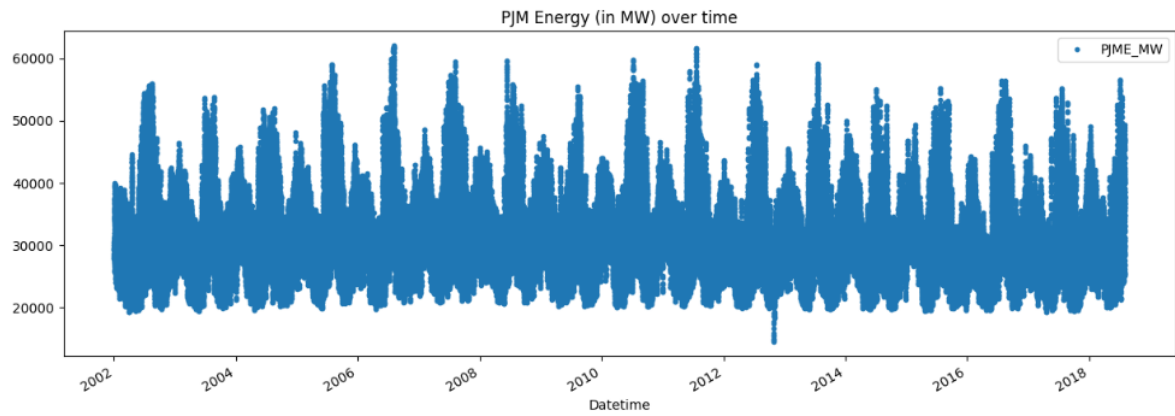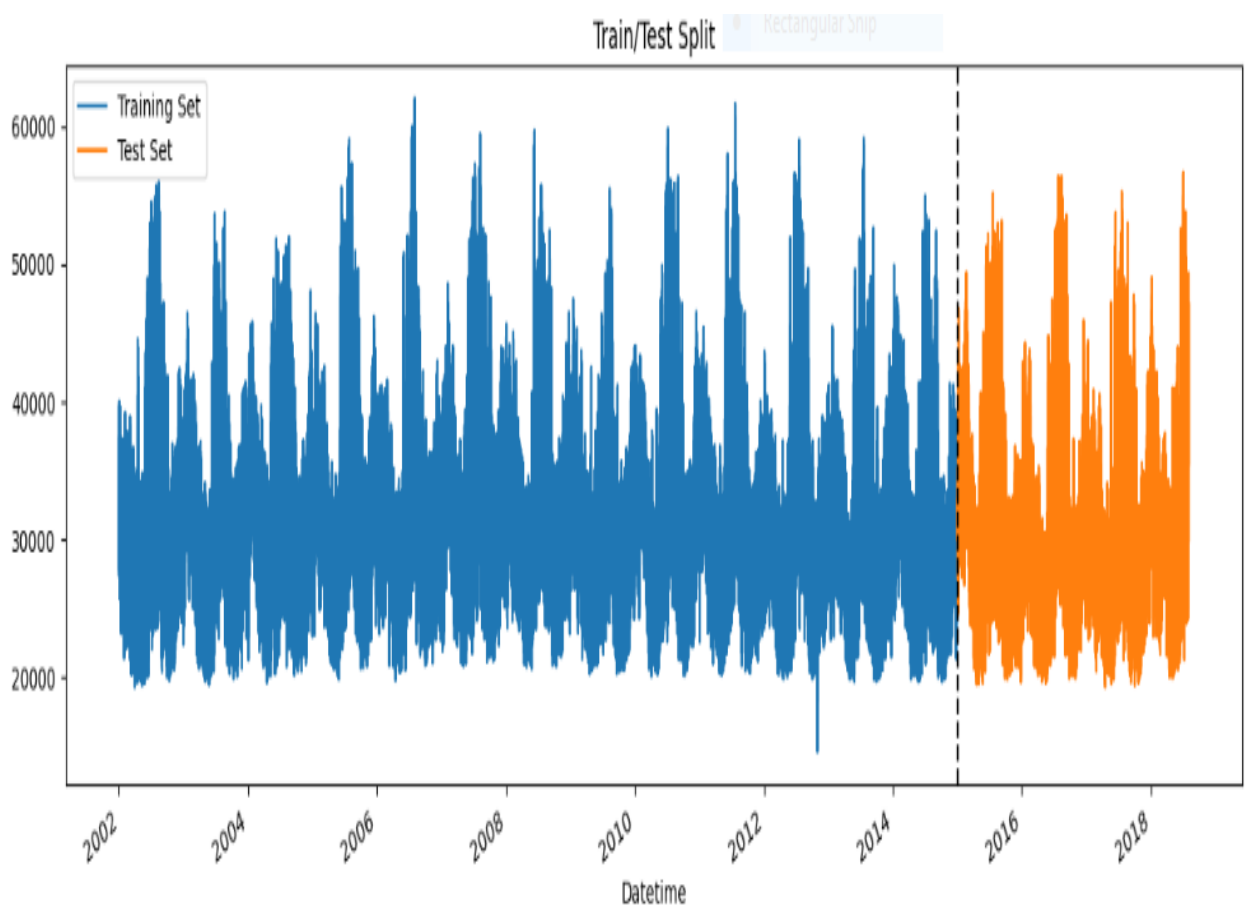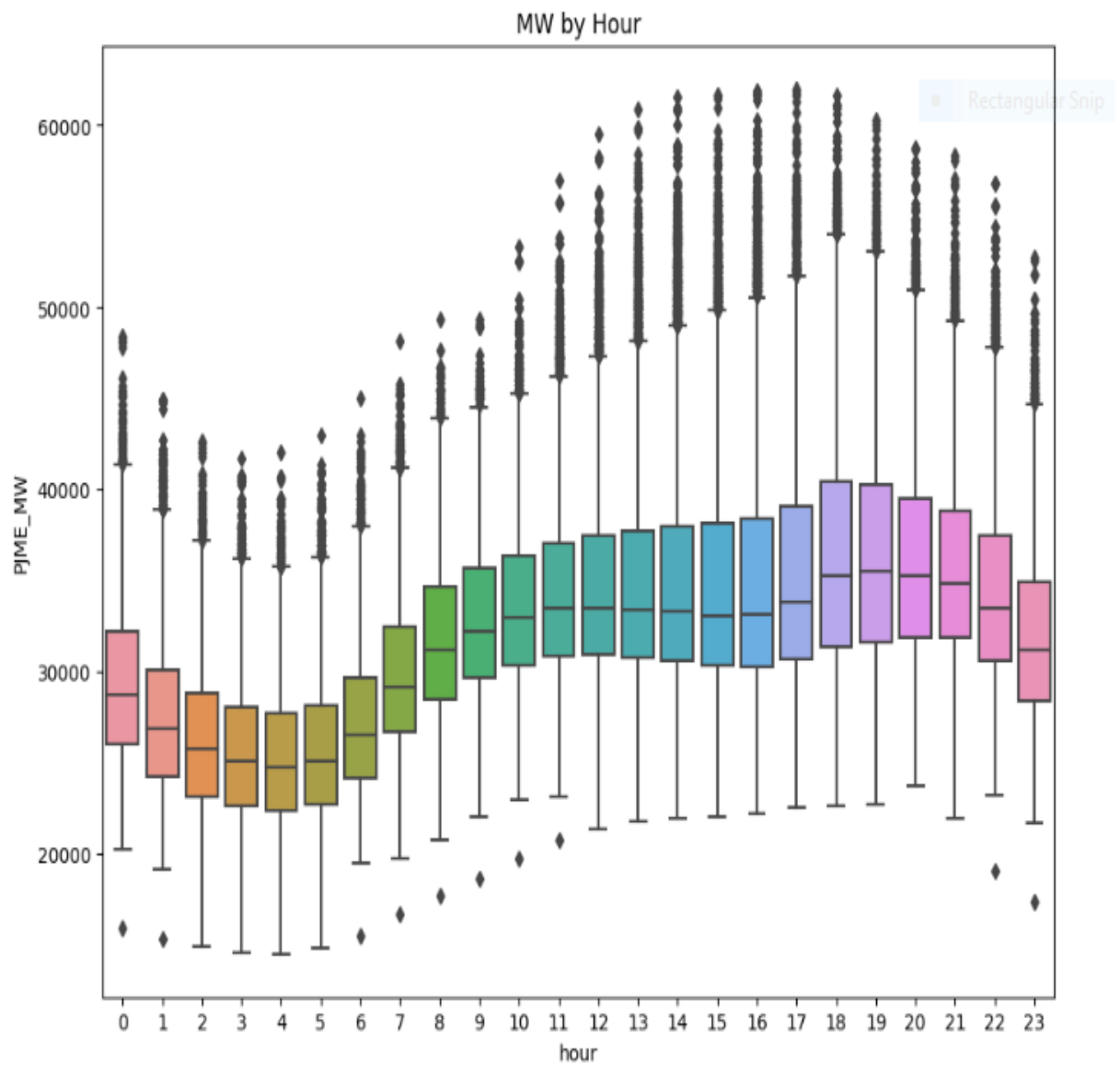
OUTPUT:

PJM Energy (in MW) over time

OUTPUT:

Train/Test Split

OUTPUT:

MW by Hour

OUTPUT:

MW by Month
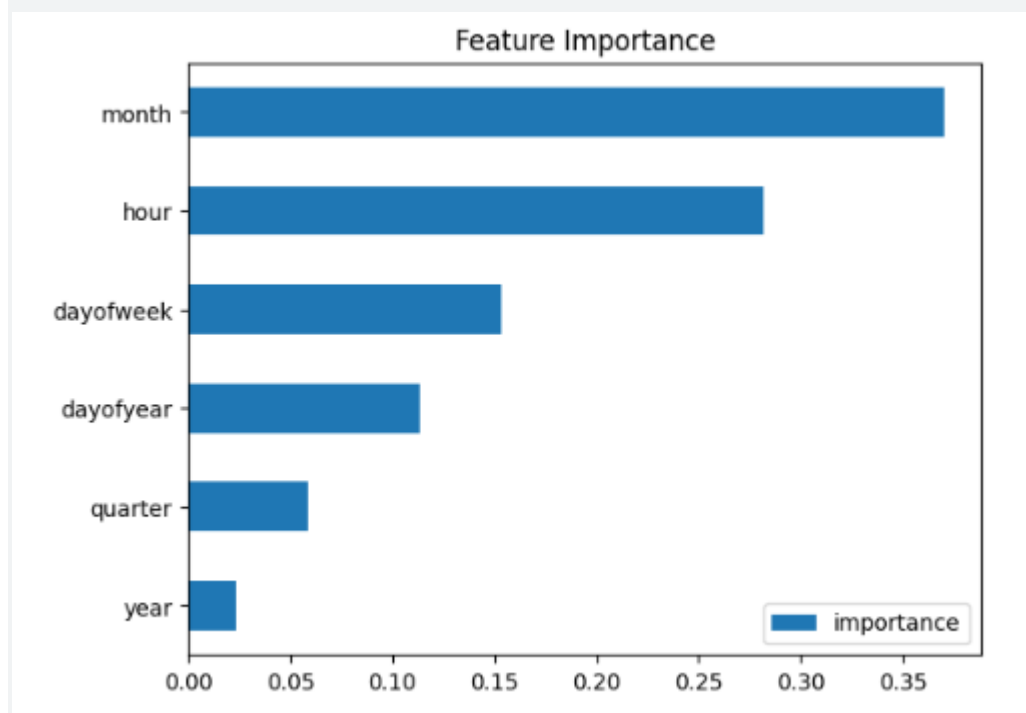
OUTPUT:

Feature Importance
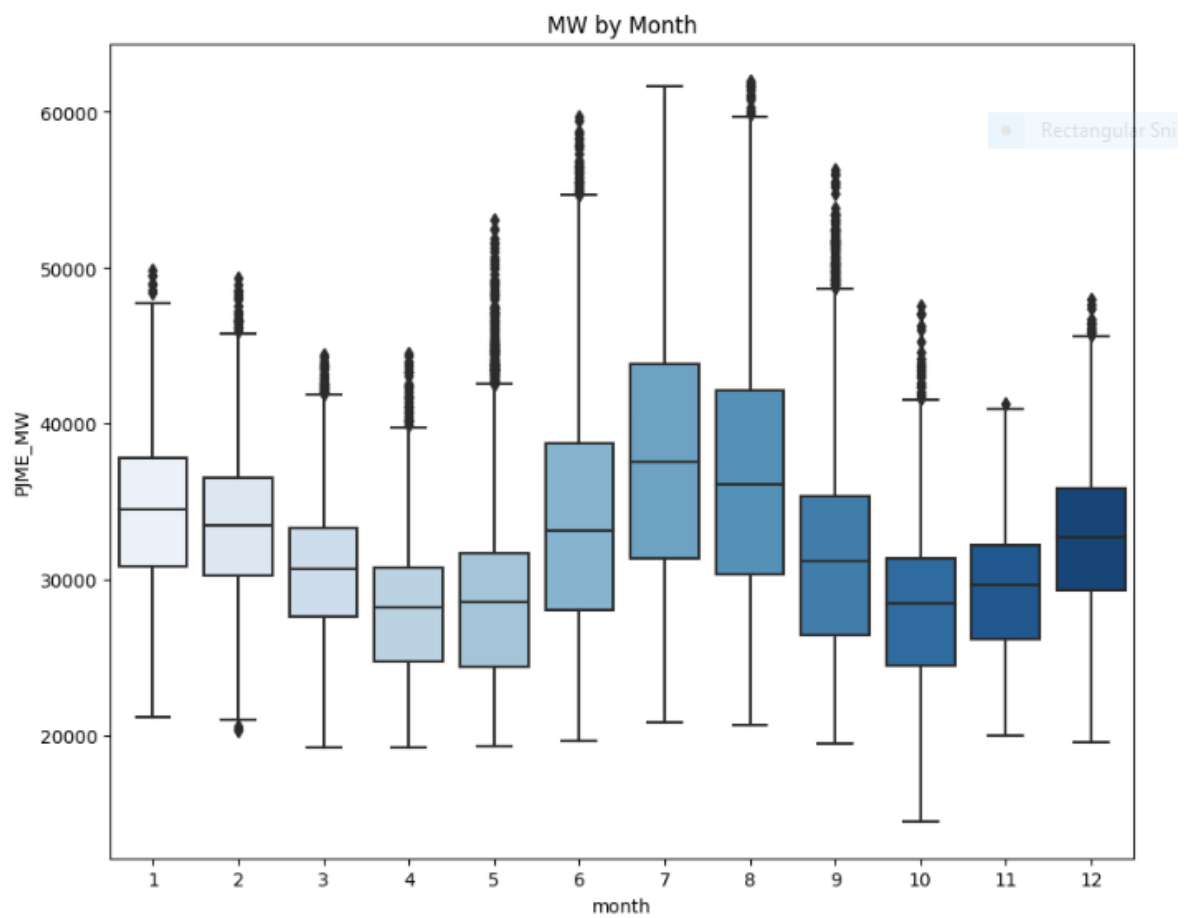
CONCLUSION:

In conclusion, the development of an automated energy consumption monitoring system with data analysis and visualization is a complex and multifaceted task. It encompasses various stages, including data collection, storage, analysis, visualization, and user interface development. Ensuring data security, scalability, and reliable automation further add to the complexity. Regular maintenance and updates are necessary to keep the system accurate and secure. This endeavor demands a multidisciplinary team with expertise in hardware, software, data science, and user interface design.