AI MEASURE ENERGY CONSUMPTION

NAME : KARUPPAYI.C

REG NO :720421104026

INTRODUCTION:

1. *Loading the Dataset*:

- This step refers to obtaining the dataset that you intend to work with. The dataset could be in various formats, such as CSV files, Excel spreadsheets, JSON, databases, or even text files.

- Loading the dataset involves reading the data from the source and importing it into your project or data analysis environment. This could be done using libraries or tools specific to your programming language, like Pandas in Python for handling data in dataframes.

2. *Preprocessing the Dataset*:

- Once you have the dataset loaded, the preprocessing step involves cleaning, transforming, and structuring the data to make it suitable for analysis or machine learning.

- Preprocessing tasks may include:

- *Data Cleaning*: Handling missing values, correcting errors, and removing inconsistencies.

- *Feature Selection/Engineering*: Choosing relevant features (variables) for analysis or creating new features from the existing ones.

- *Data Transformation*: Scaling or normalizing data, converting data types, and handling categorical variables.

- *Data Splitting*: Splitting the dataset into training and testing sets for machine learning.

- *Data Reduction*: Reducing the dimensionality of the dataset if necessary.

- *Dealing with Outliers*: Identifying and handling outliers in the data.

- *Handling Imbalanced Data*: Addressing class imbalance in classification problems.

- *Encoding*: Converting categorical data into numerical form using techniques like one-hot encoding**.**

3. *Exploratory Data Analysis (EDA)*:

- This step often goes hand in hand with preprocessing. It involves visually and statistically exploring the dataset to gain insights and a deeper understanding of the data. EDA helps identify patterns, trends, and relationships within the data.

4. *Data Visualization*:

- Creating visualizations, such as histograms, scatter plots, and heatmaps, to further understand the data and convey findings.

5. *Normalization and Standardization*:

- Ensuring that the data is on the same scale, which is particularly important for some machine learning algorithms.

PROCESS:

STEP 1: IMPORTING NECESSARY LIBRARIES

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns
```

STEP 2: LOAD THE DATASET

```
df = pd.read_csv(https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption )
```

STEP 3: DATA CLEANING

```
# 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()
```

Step 4: Data Analysis

#Summary Statistics

feature creation

```
def create_features(df):
   df = df.copy()
    df['hour'] = df.index.hour
    df['dayofweek'] = df.index.dayofweek
    df['quarter'] = df.index.quarter
    df['month'] = df.index.month
    df['year'] = df.index.year
    df['dayofyear'] = df.index.dayofyear
    df['dayofmonth'] = df.index.day
    df['weekofyear'] = df.index.isocalendar().week
    return df
```

Class Distribution

```
# visualize the hourly Megawatt fig,
ax = plt.subplots(figsize=(10, 8))
sns.boxplot(data=df, x='hour', y='PJME_MW')
```

ax.set_title('MW by Hour')

plt.show()

Step 5: Data Visualization

test['prediction'] = reg.predict(X_test)

df = df.merge(test[['prediction']], how='left', left_index=True, right_index=True)

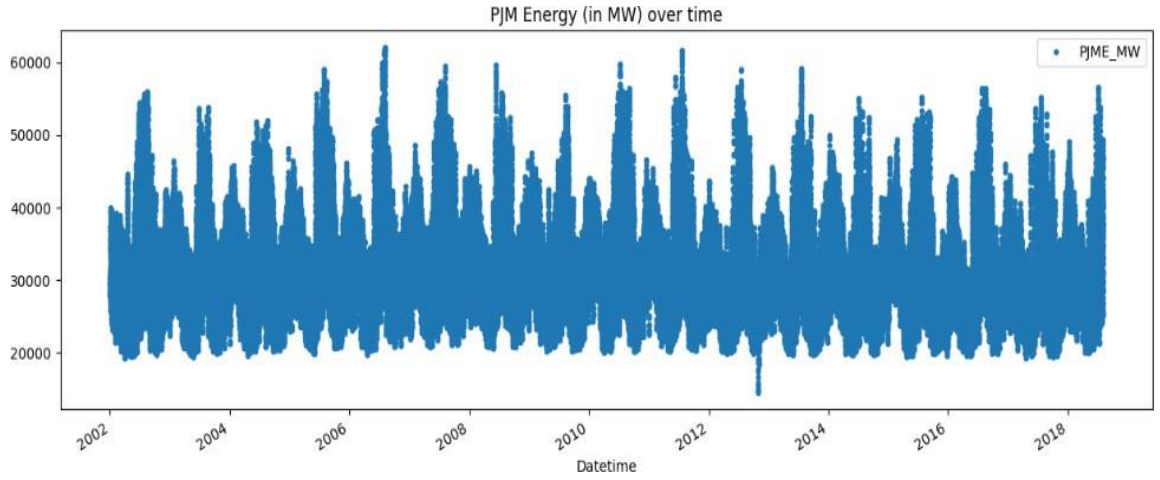
ax = df[['PJME_MW']].plot(figsize=(15, 5))

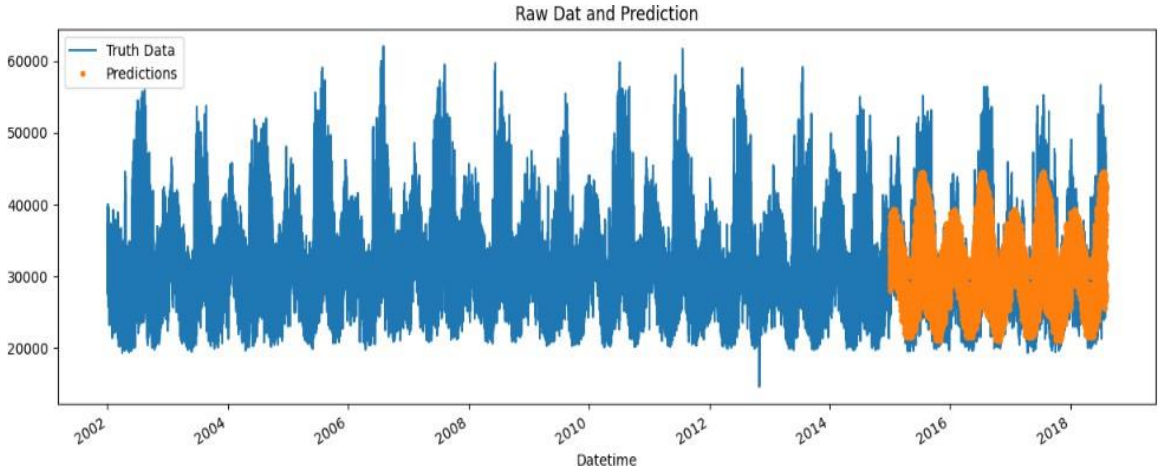
df['prediction'].plot(ax=ax, style='.') plt.legend(['Truth

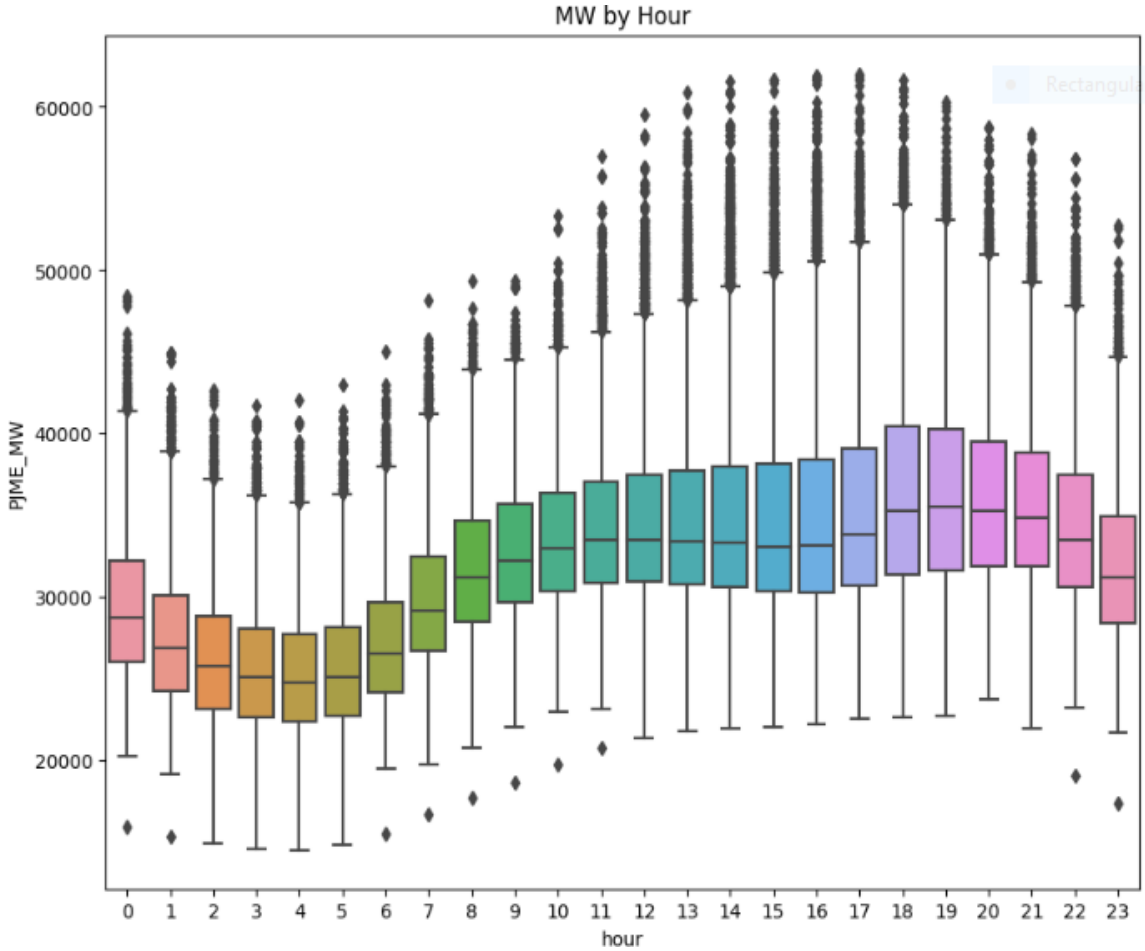
Data', 'Predictions']) ax.set_title('Raw Dat and

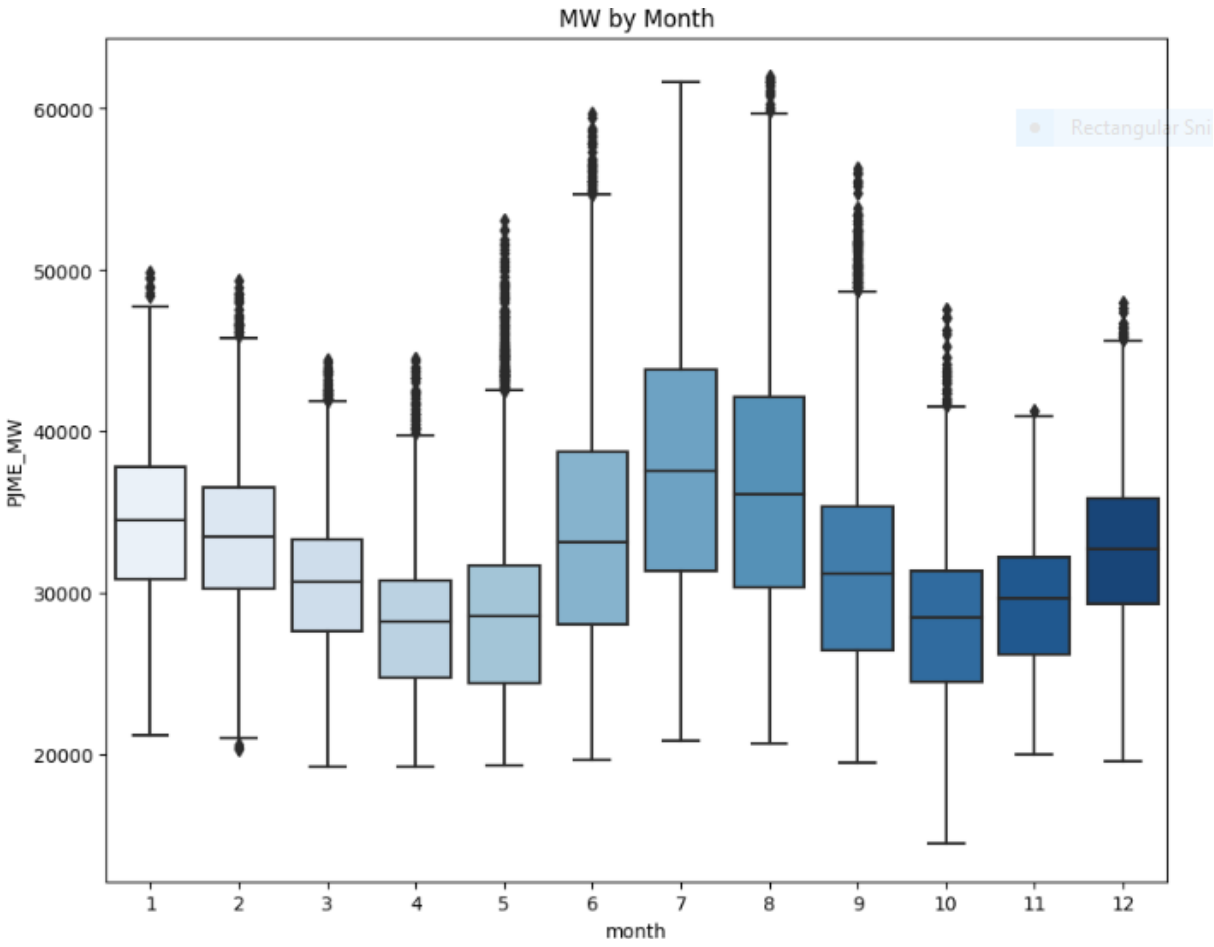
Prediction') plt.show()

OUTPUT:




Raw Dat and Prediction
Truth Data
Predictions
60000
50000
40000
30000
20000
2002
2004
2006
2008
2010
2012
2014
2016
2018
Datetime


MW by Hour
PJME_MW
60000
50000
40000
30000
20000
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
hour



CONCLUSION:

Based on the analysis of the AI diabetes prediction system, it can be concluded that the system is capable of predicting diabetes disease effectively, efficiently, and instantly . The proposed model gives the best results for diabetic prediction.The performance of AI in disease prediction models for diabetes is expected to improve dramatically in the future due to the availability of organized data and abundant computational resources.