hourly energy consumption

Abstract

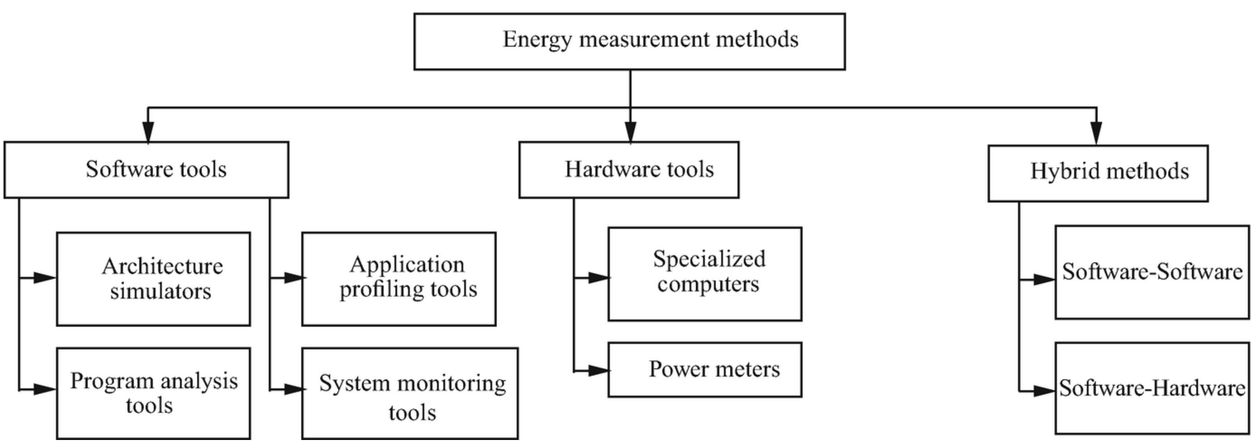
Energy consumption has been widely studied in the computer architecture field for decades. While the adoption of energy as a metric in machine learning is emerging, the majority of research is still primarily focused on obtaining high levels of accuracy without any computational constraint.

Introduction

Computer architecture researchers have been investigating energy consumption for decades, especially to be able to deliver state-of-the-art energy efficient processors. Machine learning researchers, on the other hand, have been mainly focused on producing high accurate models without considering energy consumption as an important factor [18]. This is the case for deep learning, where the goal has been to produce deeper and more accurate model without any constraints in terms of computation.

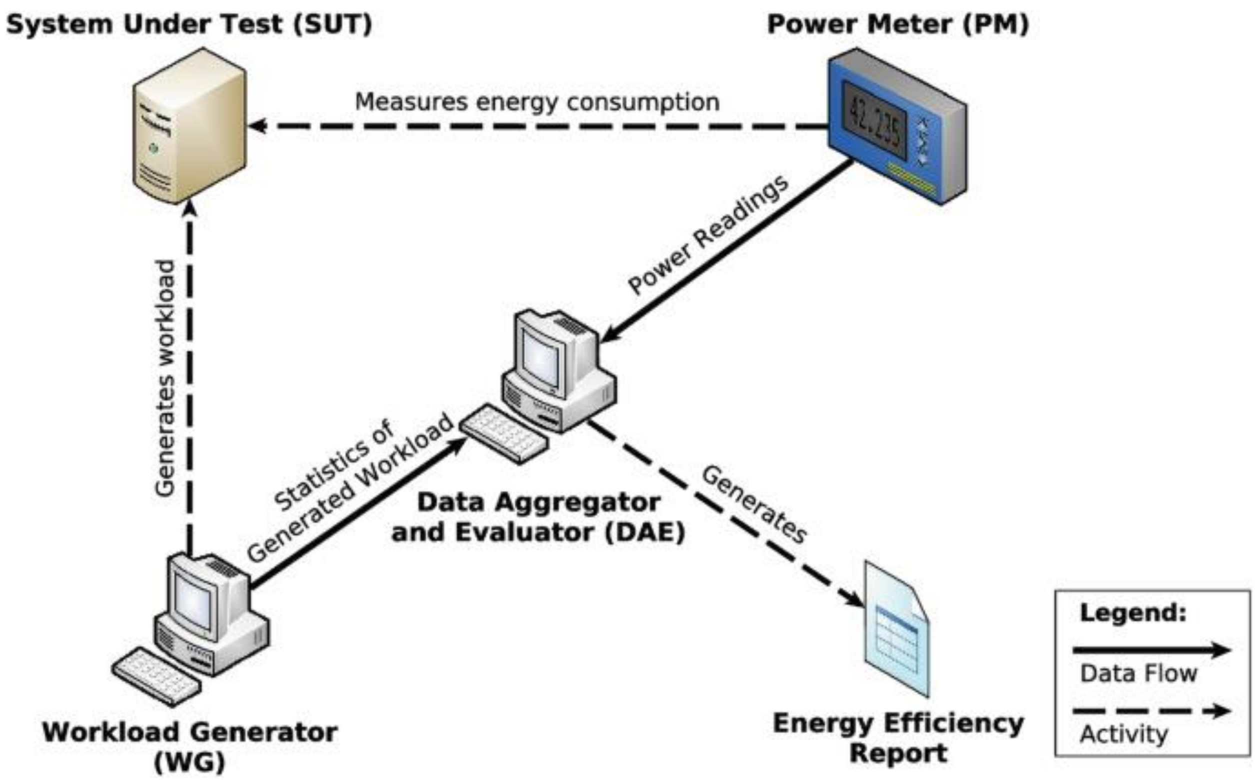
Energy Measurements Methods

Over the years, researchers and engineers have developed hardware and software tools to measure power consumption. Figure 3.1 presents a taxonomy of these developed tools and software for power consumption measurement.



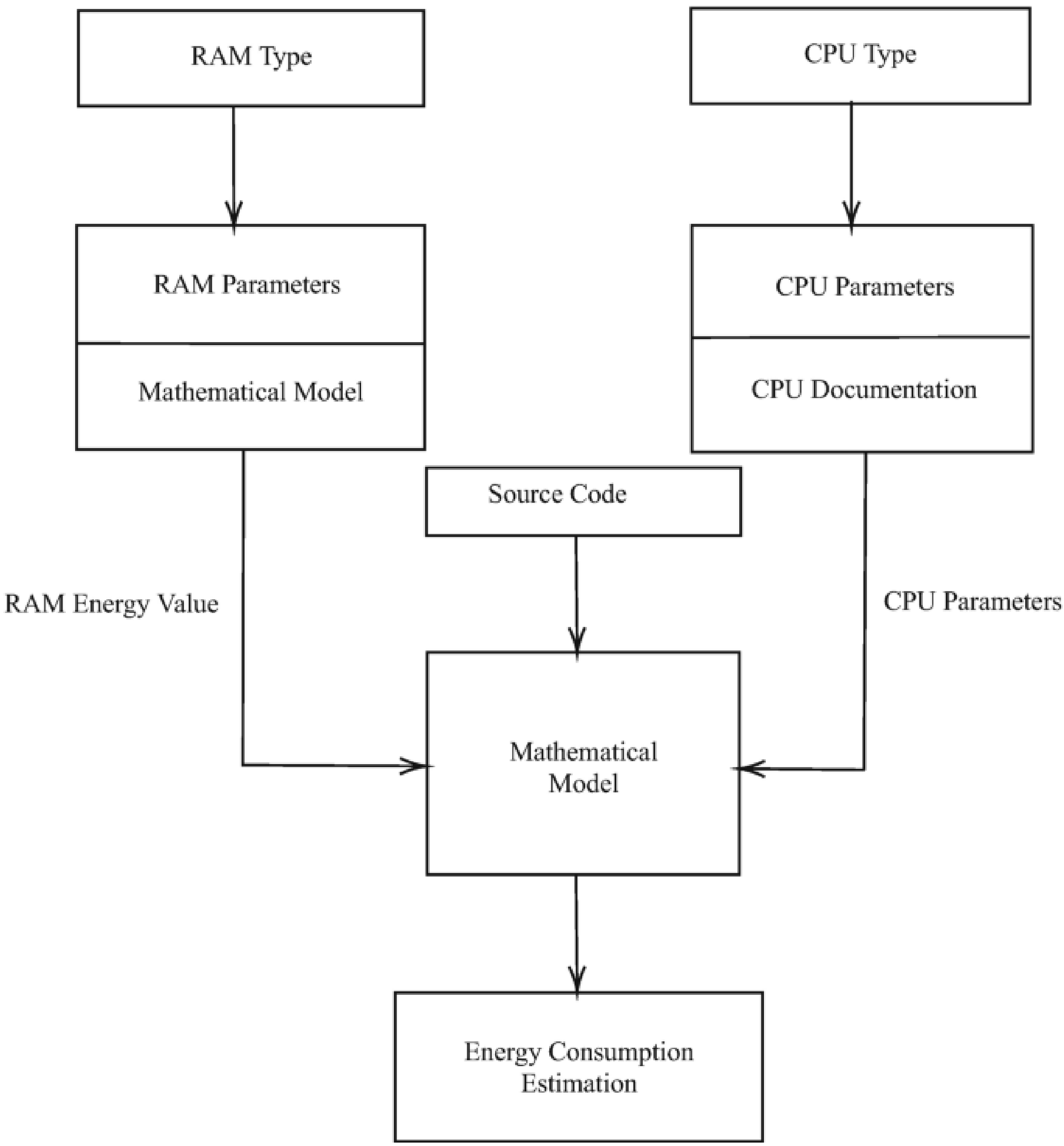
Hardware Tools

Physical power monitors are the most accurate tools to measure the energy consumed by any device connected to an electric socket. Power monitors are directly connected to the power source of the device and measure the actual power leveraged at any instant of time. Despite the precision of the approach to measure the energy consumed by a system, it becomes unfeasible when scaled up



Software Tools

Software tools include software programs that estimate the system components' or the whole system energy consumption. Over the years, researchers have developed mathematical models that determine the absolute level of power, which is necessary to transfer 1 Bit of Random Access Memory (RAM) from 0 to 1. Figure 3.4 presents a simple example of how the majority of software tools estimate energy consumption based on mathematical models.



Program

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
```

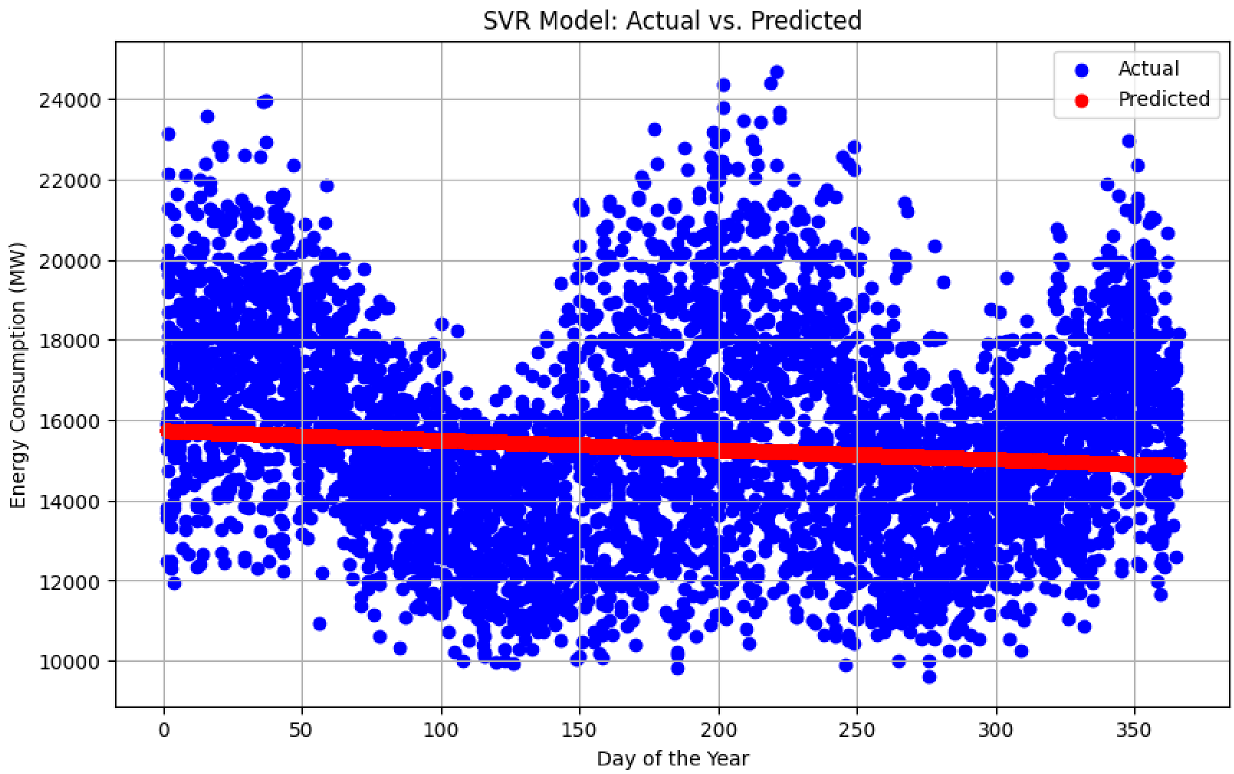
```
warnings.filterwarnings("ignore", category=UserWarning)

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, r2_score

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 + "\nDATA 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 + "\nDATA ANALYSIS" + RESET)
# --- Summary Statistics
summary_stats = df.describe()
print(GREEN + "Summary Statistics : " + RESET)
print(summary_stats)

# SUPPORT VECTOR MODELLLING
print(BLUE + "\nMODELLING" + RESET)
# Reduce the dataset size for faster training
df = df.sample(frac=0.2, random_state=42)
# Split the data into features (Datetime) and target (AEP_MW)
X = df[["Datetime"]]
y = df["AEP_MW"]
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42 )


# Preprocess the features (Datetime) to extract the day of the year
```

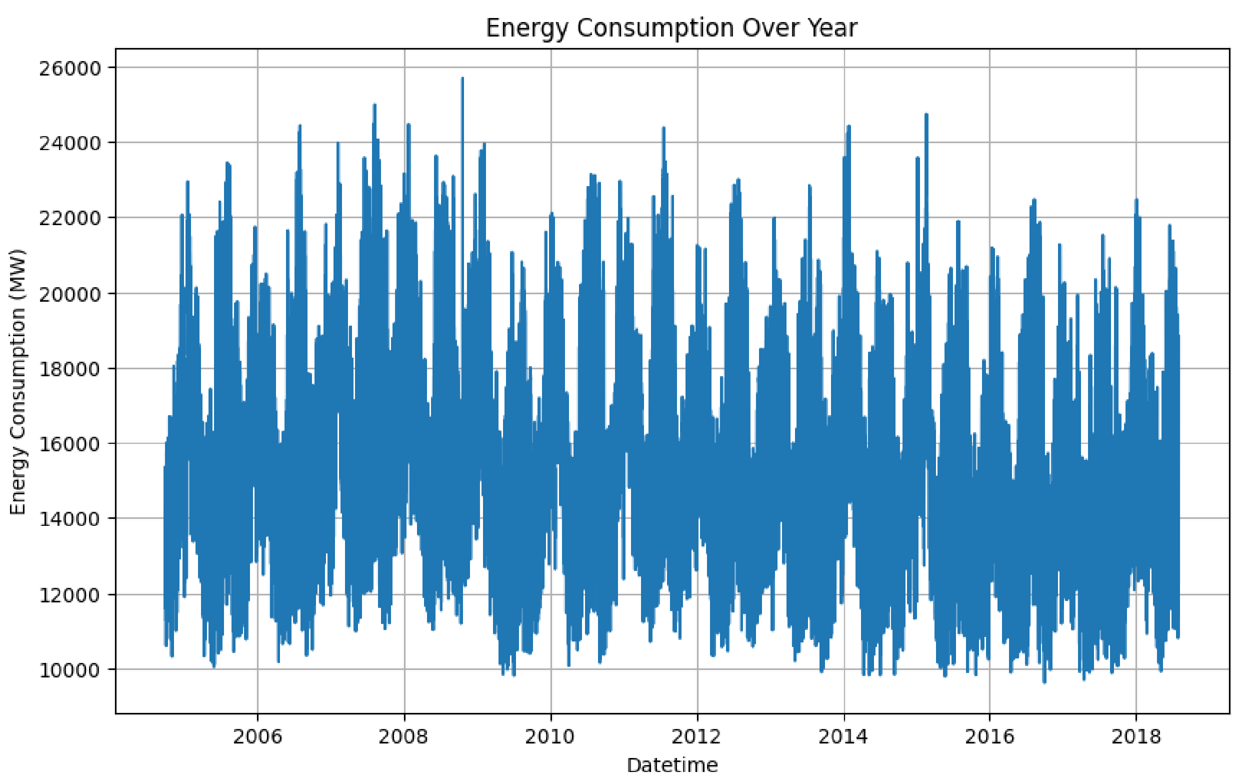
```python
X_train["DayOfYear"] = X_train["Datetime"].dt.dayofyear
X_test["DayOfYear"] = X_test["Datetime"].dt.dayofyear
# Convert X_train and X_test to NumPy arrays
X_train = X_train["DayOfYear"].values.reshape(-1, 1)
X_test = X_test["DayOfYear"].values.reshape(-1, 1)
# Standardize the data
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
# Create an SVR (Support Vector Regression) model with a linear kernel
svr = SVR(kernel="linear", C=1.0)
# Train the SVR model
svr.fit(X_train_scaled, y_train)
# Predict on the test set
y_pred = svr.predict(X_test_scaled)
# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
# Plot the actual vs. predicted values
plt.figure(figsize=(10, 6))
plt.scatter(X_test, y_test, color="b", label="Actual")
plt.scatter(X_test, y_pred, color="r", label="Predicted")
plt.xlabel("Day of the Year")
plt.ylabel("Energy Consumption (MW)")
plt.title("SVR Model: Actual vs. Predicted")
plt.legend()
plt.grid()
plt.show()

# DATA VISUALIZATION
print(BLUE + "\nDATA 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")
```

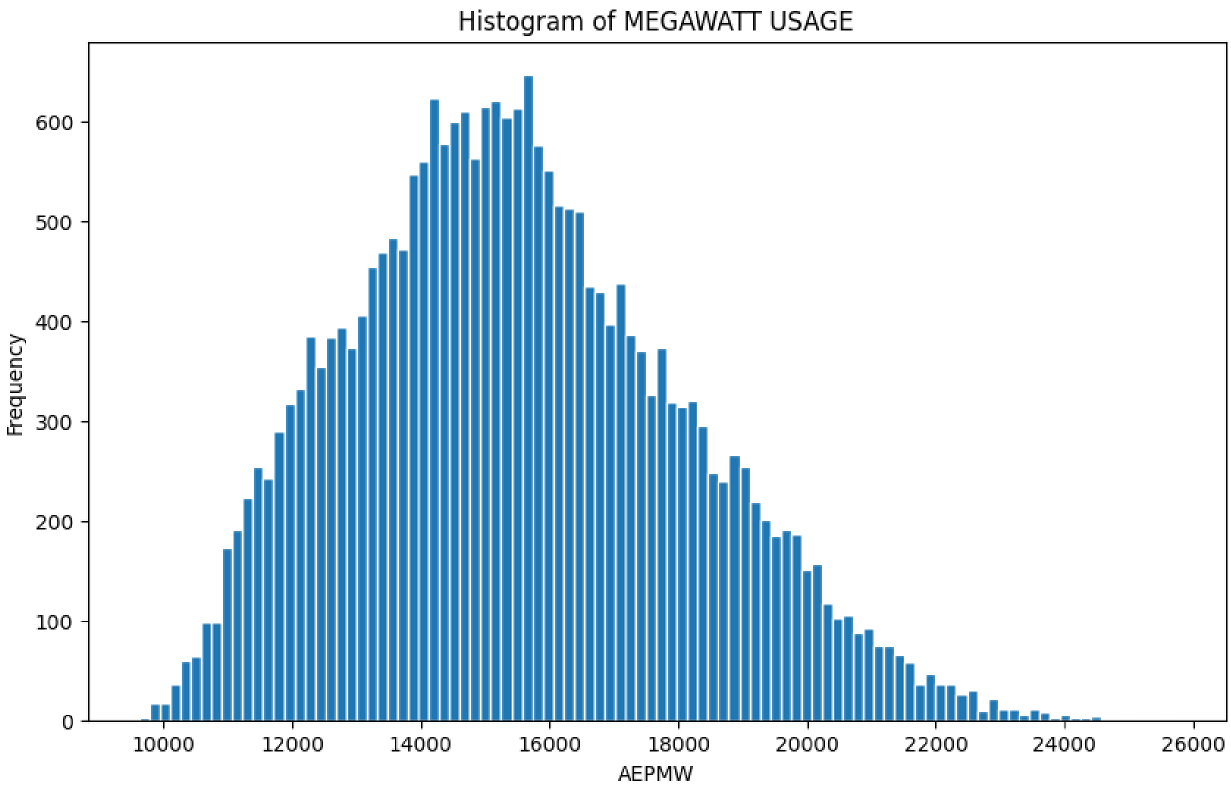
```python
plt.ylabel("Frequency")
plt.title("Histogram of MEGAWATT USAGE")
plt.show()
```

```
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

MODELLING
Mean Squared Error: 6758395.805638685
R-squared: 0.00270160624748228
```


SVR Model: Actual vs. Predicted
Actual
Predicted
Energy Consumption (MW)
Day of the Year
10000
12000
14000
16000
18000
20000
22000
24000
0
50
100
150
200
250
300
350


Energy Consumption Over Year
Energy Consumption (MW)
Datetime
10000
12000
14000
16000
18000
20000
22000
24000
26000
2006
2008
2010
2012
2014
2016
2018



Conclusion

The presented results provide a starting point for the research community as it tries to accurately estimate the amount of energy consumed in software development leveraging the benefits of ML. However, there is still room for improvement in the proposed approach above. The challenges faced by the research community have been presented and the existing energy consumption measurement tools have also been discussed.