NATIONAL INSTITUTE OF TECHNOLOGY PATNA

A PROJECT REPORT On

"ENERGY PREDICTION IN SMART BUILDING"

B. TECH CSE

SUBMITTED BY

- NAMALA VENKATARAO (2006014)
- YALLAMPATI HEMAVARDHINI (2006074)
- KETHAVATH SUNDAR (2006099)

UNDER THE SUPERVISION OF
Dr. Anshul Sharma (Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,
NATIONAL INSTITUTE OF TECHNOLOGY PATNA

May 2023

CERTIFICATE

Certified that this project report "ENERGY PREDICTION IN SMART BUILDING using FORECASTING MODELS PROJECT" the bonafide work of "Mr. NAMALA VENKATARAO (2006014), Miss. YALLAMPATI HEMAVARDHINI (2006074), Mr. KETHAVATH SUNDAR (2006099)" who carried out the project work under my supervision, during the academic year 2023. It is certified that all the corrections/suggestions indicated for Internal Assessment have been incorporated in the project report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed**.**

Name of Supervisor: Dr. Anshul Sharma

Signature:

Signature of Head of the Department (CSE)

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of the people who made it possible, success is the epitome of hard work and perseverance, but steadfast of all is encouraging guidance. So, it is with gratitude that we acknowledge all those whose guidance and encouragement served as a beacon of light and crowned our effort with success.

We would like to express my special thanks of gratitude to our project supervisor **Dr. Anshul Sharma** who gave us the golden opportunity to do the project **Energy prediction** using Forecasting, who also helped us in doing a lot of research and we came to know about so many new things. We are really thankful to him.

Secondly, we would also like to thank our parents and friends who helped us and provided us with resourceful information and motivations which helped a lot in finalizing this project within the limited time frame.

ABSTRACT

This study details the public release of power usage and interior environmental measurement data from a seven-story, 11,700-square-meter office building in Bangkok, Thailand.

Each of the building's 33 zones has its own breakdown of power use (in kilowatt hours) for things like air conditioning, lights, and plug loads. Measurements of temperature, humidity, and lux were taken in the identical rooms using indoor environmental sensors. From July 1, 2018 until December 31, 2019.

you may get the full datasets every minute. In addition to helping with tasks like load forecasting at the zone, floor, and building levels, creating indoor thermal models, validating building simulation models, creating demand response algorithms by load type, detecting anomalies, and controlling multiple air conditioners with reinforcement learning algorithms all benefit from access to such data.

It is implemented using simple python modules such as matplotlib, Timeseries, Forecasting models and Machine learning approaches which are used to create data frames and plot different graphs.

Keywords: Power Consumption, Indoor Environmental sensors, Timeseries, Forecasting, matplotlib, jupyter notebook, Machine learning approaches (LSTM, XG Boost) etc.

Subject terms: Energy management, Energy efficiency, Energy and behaviour

Measurement(s)	electrical energy • temperature of air • humidity • visible spectrum radiation
Technology Type(s)	Gauge or Meter Device • Sensor Device
Factor Type(s)	floor • date/time
Sample Characteristic - Environment	building

TABLE OF CONTENTS

1. Introduction

The first step in implementing a Forecasting timeseries algorithm is to understand the right learning experience from which the model starts improving on. Data pre-processing plays a major role when it comes to forecasting. The first step is pre-processing and data pre-processing plays a major role when it comes to machine learning. In order to apply the libraries, it has to be pre-processed and stored in an efficient way.

What makes the CU-BEMS dataset provided in this research stand out is its dissection of total building power use (in kW) down to the zone and floor levels. In the CU-BEMS dataset, every minute of operation is recorded for each zone's air conditioners, lights, and plug loads. All commercial structures must deal with these three enormous loads. In addition, at minute-by-minute intervals, data from interior environmental sensors (including temperature, humidity, and ambient light) are collected from each zone.

Energy Monitoring Units (EMUs), digital metres, multi-sensors, gateways, and a CU-BEMS server make up the larger CU-BEMS system. Developed in-house are the EMUs, multi-sensors, gateways, and server. There is a total of 21 EMUs, 30 digital metres, 24 multi-sensors, and 7 gateways installed as part of the CU-BEMS rollout at the Chamchuri 5 building. Individual 1-2 kW wall-mounted air conditioning units, lights, and plug load circuits may all be monitored with a single EMU. Each digital metre is calibrated to take readings from a big AC compressor (often rated at 20–40 kW). Each of these advanced sensors can detect changes in temperature, humidity, and illumination. Plans and rough pinpoints of where sensors are placed.

1.1 Objective

The objective of smart building energy prediction using forecasting models is to accurately predict the future energy consumption or demand of a building based on various factors such as weather conditions, occupancy patterns, and other relevant data. The goal is to optimize building energy management and reduce cut down on energy use, expenses, and greenhouse gas emissions.

Forecasting models can analyse historical energy consumption data and identify patterns and trends that can be used to predict future energy demand. By incorporating real-time data such as weather forecasts and occupancy information, the models can provide more accurate predictions and enable building managers to adjust energy usage in response to predicted changes in demand.

Smart building energy prediction using forecasting models can also help with energy procurement and scheduling, allowing building managers to take advantage of low-cost energy periods and avoid peak demand charges. It can also help identify potential energy efficiency improvements and inform decisions about equipment upgrades and maintenance.

Overall, the objective of smart building energy prediction using forecasting models is to help buildings become more energy-efficient, cost-effective, and environmentally sustainable.

1.2. Motivation

In this decade Commercial and residential buildings together account for around 20%1 of worldwide energy consumption in the building industry. From 2018 to 20502, it is expected that building energy consumption would rise at an annual pace of 1.3% due to the fast rise in population and economic development.

The survey analysis on rising energy demand has sparked widespread global worries about its harmful influence on the environment since the construction industry is responsible for one-third of greenhouse gases, two-thirds of halo-carbon, and about 25-33% of black carbon emissions3. Technologies exist now that can cut a building's power use by as much as 80%.4.

The survey also proved that increasing demand for power necessitates a well-oiled and reasonably priced machine. Smart metres and building automation systems (BAS) are being installed more often to collect detailed information about building loads. Due to the abundance of building-level data, data-driven approaches have emerged as a viable alternative to the more conventional physics-based methods.

However, high-quality building data is crucial for studies employing data-driven methods. There are both commercial and residential building electricity databases accessible for public use in the field of building electricity research.

Our project ensures to provide an in-depth exploratory data analysis on predicting power consumptions using forecasting.

2. Methodology

2.1 Background

The expanding area of data science includes machine learning as a key component. Algorithms are acquired using statistical techniques to produce classifications or predictions and to find important insights in data science projects. The decisions made as a result of these insights influence key growth indicators in applications and enterprises, ideally. Data scientists will be more in demand as big data continues to develop and flourish. They will be expected to assist in determining the most pertinent business issues and the information needed to address them.

Time series forecasting, which involves predicting future events based on historical data, is a crucial tool with a broad range of applications such as forecasting product demand, weather, and healthcare trends. The two primary methods for time series forecasting are statistical approaches and neural network models. While statistical methods like ARIMA, ARIMAX and SARIMAX are popular, this article explores the use of neural network models like LSTM, specifically extreme learning machines, for time series forecasting.

Extreme Learning Machine's (ELM) can be a highly effective tool for time series forecasting. While statistical approaches like ARIMA models are limited to stationary time series, neural network models can capture non-linearities caused by external factors. However, traditional backpropagated NNs have drawbacks such as slow learning and overfitting. ELMs, on the other hand, offer fast learning, superior generalization, and minimal need for human intervention.

Technical validation: The project started off with basic technical validation, data visualisation to demonstrate the reliability and validity of the dataset, including the presentation of missing data, histograms, and weekly trend charts. From that Missing data shows data availability. Data histogram charts show floor/zone metrics. Weekly trends demonstrate the link between AC operation, interior temperature/humidity, lighting/plug load operation, and ambient light.

2.2 Proposed models

These various modules provide better user understandability and code representation. The following libraries are used such as NumPy, pandas, matplotlib, seaborn etc.

It analyses the data from past energy consumption of smart building and predict the future power consumption by zone-floor-building level, for that implementation using forecasting models like ARIMA, LSTM…

- **Autoregressive Integrated Moving Average** (ARIMA) models are a popular class of models used for time series forecasting. They model the relationship between past values of the time series and its future values, and can be used to make short-term or long-term predictions.

Arima Parameters

P: the quantity of historical values used in the AR model

$$y_t = c + \varphi_1 y_{t-1} + \varphi_2 y_{t-2} + \dots + \varphi_p y_{t-p} + \varepsilon_t$$

where,

c=constant

$\varphi_1, \dots, \varphi_p$ = parameters

ε_t =white noise

d: the quantity of differences in the time series

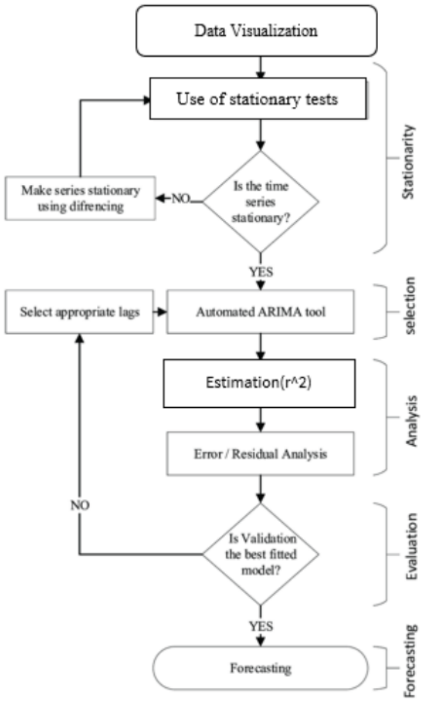
$$\nabla y_t = y_t - y_{t-1}$$

q: the MA model's inclusion of the number of historical forecasting failures

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \dots + \theta_q \varepsilon_{t-q}$$

ARIMA Model equation:

$$\boxed{\nabla y_t = c + \varphi_1 \nabla y_{t-1} + \dots + \varphi_p \nabla y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \dots + \theta_q \varepsilon_{t-q}}$$



Flow chart 2.2a: **ARIMA MODEL**

• **Neural networks:** Deep learning models such as recurrent neural networks (RNNs) and long short-term memory (LSTM) networks can be used for time series forecasting. These models are able to capture complex patterns in the time series data and can be highly accurate.

All RNN take the shape of a series of neural network repeating modules. This recurring module in typical RNNs will be made up of just one tanh layer

Step 1: Determine How Much Historical Data It Should Remember
Choosing which data should be excluded from the cell at that specific time step is the first stage in the LSTM. This is decided using the sigmoid function. It computes the function while taking into account the current input xt and the prior state (ht-1).

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

f_t= forget gate
Decides which information to delete that is not important from previous time step.

Step 2: Calculate the Contribution of This Unit to the Present Situation
There are two sections in the second stratum. The tanh function and the sigmoid function are the two. It determines whether numbers to pass through in the sigmoid function (0 or 1). The tanh function weighs the values that are supplied, determining their level of significance (from -1 to 1).

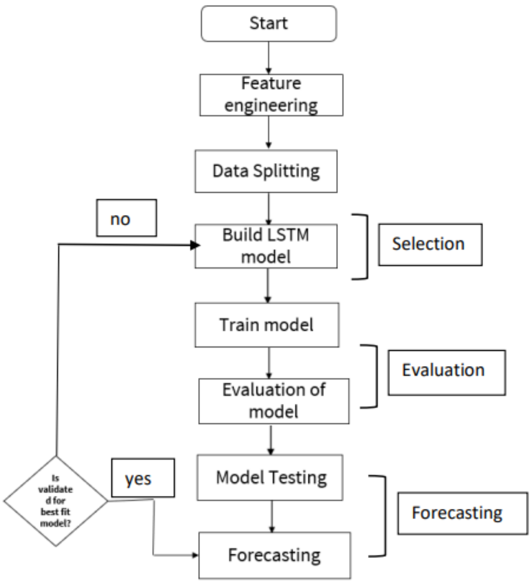
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

i_t= input gate
Determines which information to let through based on its significance in the current time step.

Step 3: Choose Which Amount of Current Cell State Is Included in Output
Making a decision about the output is the third phase. To determine which components of the cell state are output, we first run a sigmoid layer. Then, we multiply the cell state by the output of the sigmoid gate after pushing the values through tanh to be between -1 and 1.

$$o_t = \sigma\left(W_o \ [h_{t-1}, x_t] \ + \ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

o_t= output gate
allows the passed in information to impact the output in the current time step.



Flow chart 2.2b: **LSTM MODEL**

On the other hand, machine learning is a broader category of techniques that involve using algorithms and statistical models to learn patterns in data and make predictions or decisions. For that we proposed to implement the XG-booster algorithm

- **XG Boost** (Extreme Gradient Boosting) is a popular and powerful open-source library for gradient boosting that is widely used for machine learning tasks, such as regression, classification, and ranking. XG Boost is a powerful and flexible tool for building high-performance machine learning models.

The XGBoost tree for Regression may be built using the formulae shown below.

Step 1: Calculate the similarity scores; this aids in the tree's growth.

```
Similarity Score = (Sum of residuals)^2 / Number of residuals + lambda
```

Step 2: Determine how to partition the data by calculating the gain.

```
Gain = Left tree (similarity score) + Right (similarity score) - Root
(similarity score)
```

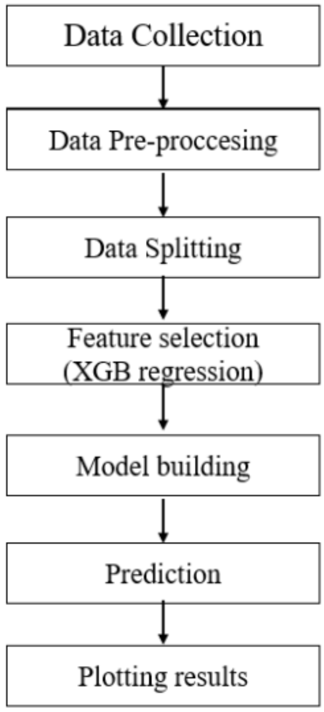
Step 3: Prune the tree using the user-defined tree-complexity parameter, gamma, to find its difference from Gain.

```
Gain - gamma
```

Step 4: Calculate the output value for the remaining leaves

```
Output value = Sum of residuals / Number of residuals + lambda
```

The similarity scores are shrunk and the output values for the leaves are less when lambda is bigger than 0. This causes more pruning. Let's look at a mathematical procedure for determining the appropriate output value to minimise the loss function. For regression and categorization



Flow chart 1.4: **XGB Regression**

3. Experimental setup

3.1 Platform

The Jupyter Notebook App is a server-client programme that enables web browser-based editing and execution of notebook papers. The Jupyter Notebook App may be used offline, without an internet connection, on a local desktop, as detailed in this paper, or it can be deployed on a remote server and accessed online.

3.1.1 System requirements

Operating systems: Windows 10 and 11 (Intel/AMD 64-bit)

- Intel CPUs are recommended, though not required. They have an optimized Intel Machine Learning library that offers performance gains for certain Machine Learning algorithms.

3.1.2 Installation

Step 1: The browser
Installing a current browser that complies with standards constitutes step "one". Google Chrome or Mozilla Firefox both perform fine. Try to avoid MS Explorer

Step 2: Installation
Installing a scientific Python installation that also includes scientific Python packages is the simplest approach to install the Jupyter Notebook App. The Anaconda distribution is the most popular:

- Download Anaconda Distribution (a few 100MB), Python 3, 64 bits.
- Install it using the default settings for a single user.

Step 3: Launching Jupyter Notebook App

- Change Jupyter Notebook startup folder (Windows)

Step 4: Executing a notebook

- Place the downloaded notebook in your folder for notebooks.
- You can run the notebook document step-by-step (one cell a time) by pressing shift + enter.
- You can run the whole notebook in a single step by clicking on the menu Cell -> Run All.

3.2 About dataset

The CU-BEMS dataset given here is a breakdown of building-level electricity usage (kW) by zone and floor. At one-minute intervals, the CU-BEMS dataset records the functioning of individual AC units, lights, and plug loads in each zone of the building. These are the three most important loads in commercial structures. Furthermore, relevant indoor environmental sensor data (temperature, humidity, and ambient light) are measured at one-minute intervals in each zone.

CU-BEMS dataset file names.

	Year 2018	Year 2019
Floor1	*2018Floor1.csv*	*2019Floor1.csv*
Floor2	*2018Floor2.csv*	*2019Floor2.csv*
Floor3	*2018Floor3.csv*	*2019Floor3.csv*
Floor4	*2018Floor4.csv*	*2019Floor4.csv*
Floor5	*2018Floor5.csv*	*2019Floor5.csv*
Floor6	*2018Floor6.csv*	*2019Floor6.csv*
Floor7	*2018Floor7.csv*	*2019Floor7.csv*

Fig 3.2.a Dataset files

The full dataset is partitioned into 14 comma-separated value (csv) files based on the floor and year of data collection. This is due to the ease with which CSV files may be imported into spreadsheets, databases, or computer languages, making it easier and more organised to deal with. Please keep in mind that one CSV file is given for each level of the building. This amounts to seven CSV files for each year. Because each file contains data for each zone on a particular floor, a user may work with any specific zones that can be retrieved (based on the column names) from the CSV files.

Each of the 2018 data files has 264,960 rows that represent 1,440 data points each day at one-minute intervals for 184 days in the second half of 2018. Each of the 2019 data files contains 525,600 rows, or 1,440 data points every day for the whole year of 2019, at one-minute intervals.

3.3 Data Pre-processing

Data pre-processing plays a major role when it comes to forecasting. The first step is pre-processing and data pre-processing plays a major role when it comes to machine learning. In order to apply the libraries, it has to be pre-processed and stored in an efficient way.

- Loading of data in Pandas.
- Drop useless columns.
- Drop rows with missing values.
- Create dummy variables.
- Take care of missing data.
- Convert the data frame to NumPy.
- Divide the data set into training data and test data.

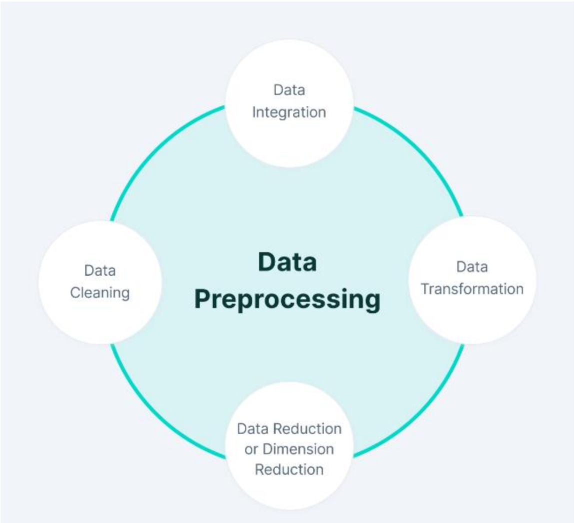
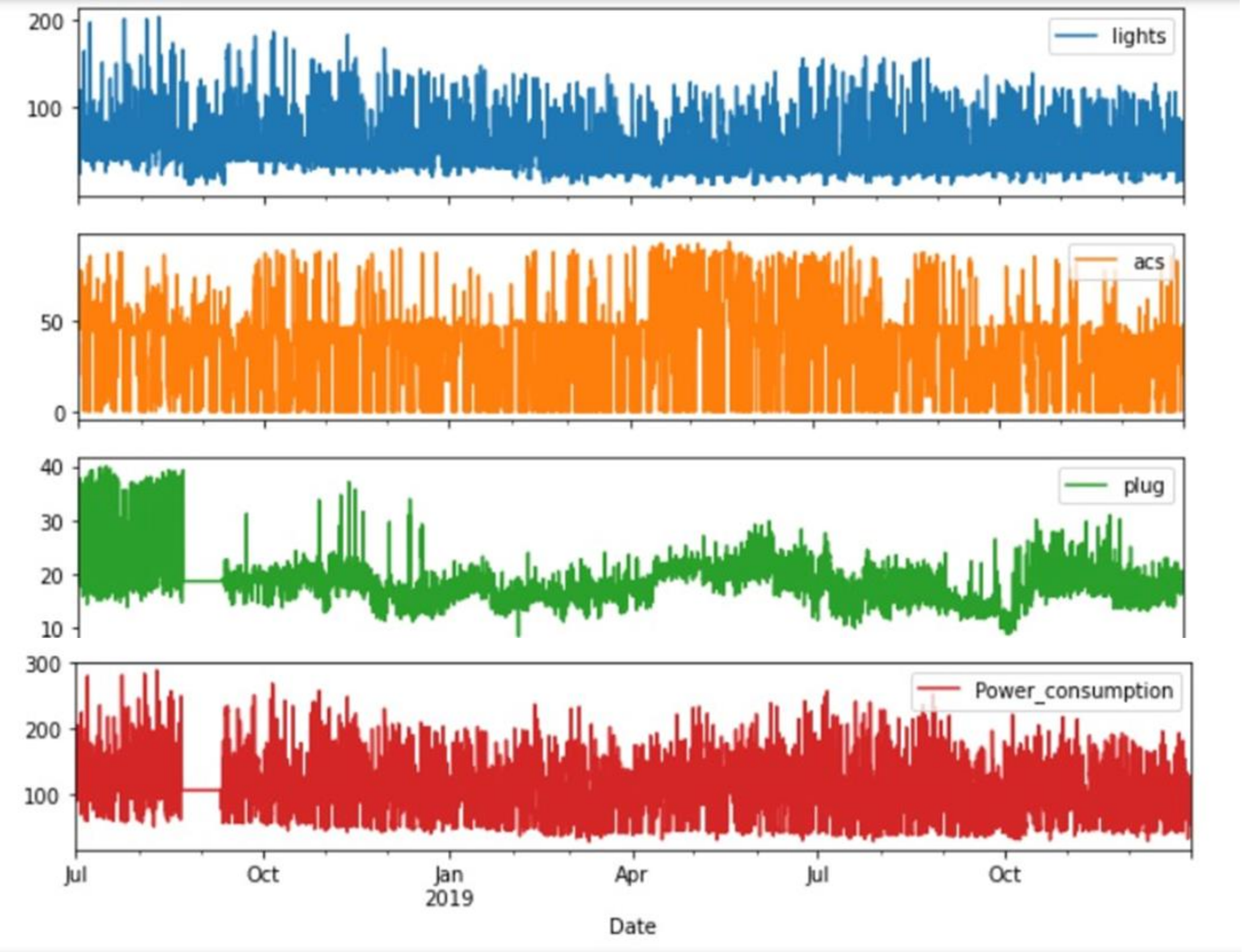


Fig 3.3a: **Data Pre-processing**

4. Result

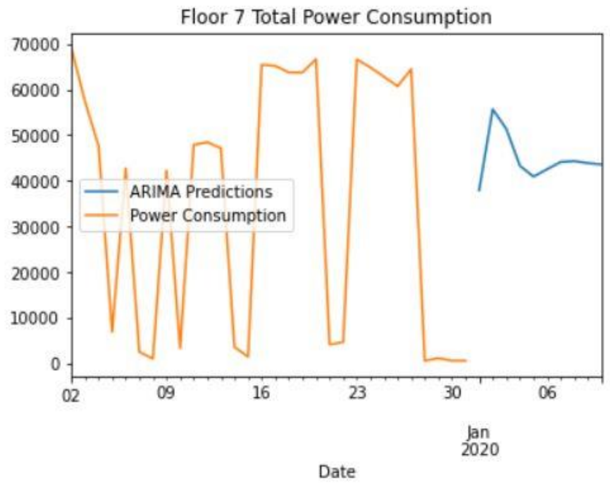
4.1 Data visualization:

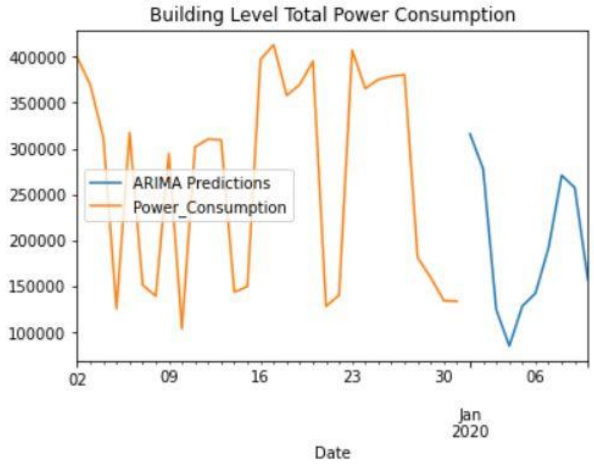


Graph 4.1a: Visualization of power consumption for individual lights, plugs & AC

4.2 Model results

Using **ARIMA** Model:

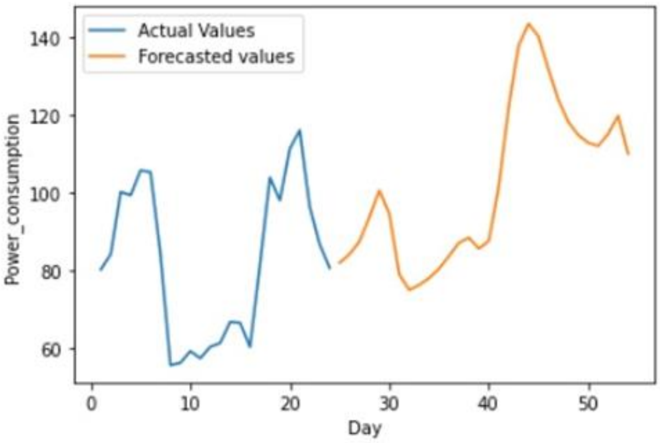


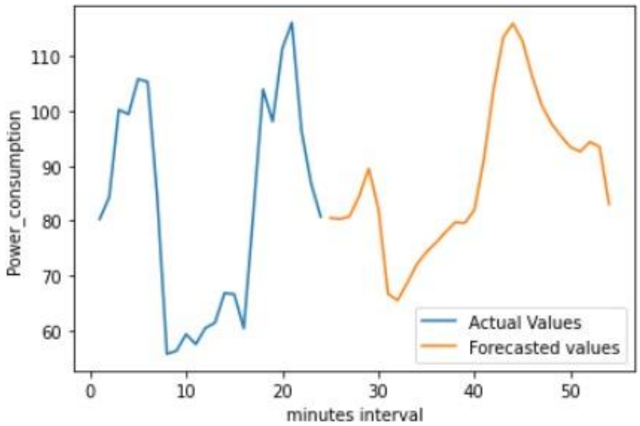


Floor-level Total Energy consumption *Building level energy consumption*

ARIMA					
Lags	**Parametres**	**RMSE**	**MSE**	**MAE**	**Accuracy**
• 10min • 20 min	• p=5 • d=1 • d=5	• 4.5329	• 19.9613	• 6.7382	• 72.452
• 10 days • 20 days	• p=5 • d=1 • d=5	• 10.1421	• 102.4621	• 7.8562	• 69.563

Using **LSTM** Model:

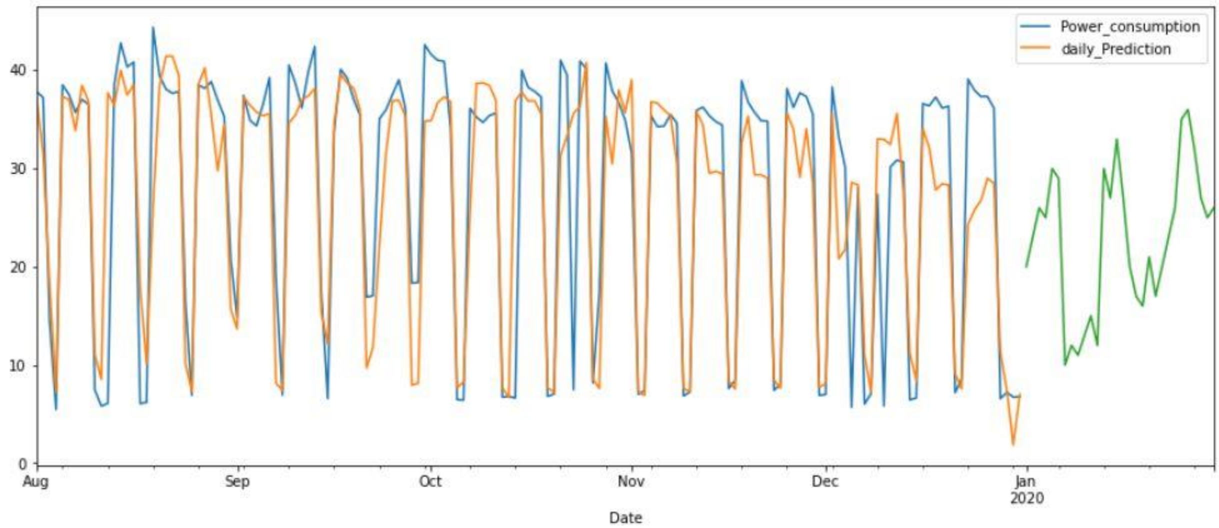




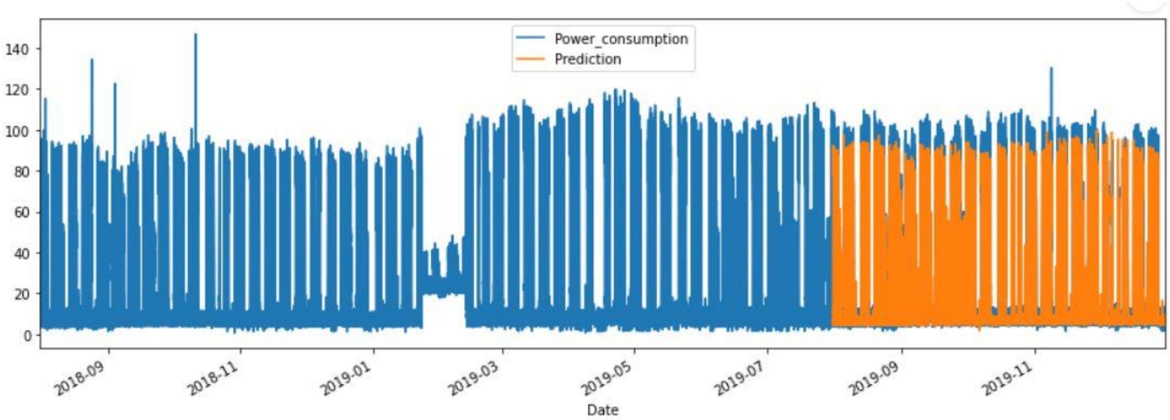
Building level total energy prediction on daily and minute basis

LSTM					
Lags	**Parameters**	**RMSE**	**MSE**	**MAE**	**Accuracy**
• 15 min • 2 hours	• input shape =1 day • Window size= 4 • no. of layers=128 • reLU (alpha=0.5)	• 4.3251	• 20.6315	• 2.7518	• 70.86
• 1 days • 5 days	• input shape =1 day • Window size=4 • no. of layers=128 • reLU (alpha=0.5)	• 9.71123	• 102.5496	• 3.9421	• 67.92

Using **XGB** Regressor:



Building level total energy prediction (Daily basis)



Building level total energy prediction (Minute basis)

XGBoost Regressor					
Lags	**Parametres**	**RMSE**	**MSE**	**MAE**	**Accuracy**
• 15 min • 30 min • 2 hours	• n_estimators=1000 • Learning rate=0.02	• 1.9369	• 6.9603	• 3.7518	• 94.86
• 10 days • 20 days	• n_estimators=1000 • Learning rate=0.01	• 2.1237	• 7.3267	• 4.5104	• 65.48

4.3 Discussion regarding the outcomes

- We are predicted future energy in smart building based on past data by using forecasting algorithms. It predicted the one month future energy consumption
- We have implemented the ARIMA ,LSTM and XGB regressor algorithms on the dataset we got overall best accuracy in XGB regressor is approximately 95 percentage by using different lag times and by changing parameters.
- We have also got least root mean squared error, mean squared error as well as mean absolute error on XGB regressor compared to remaining algorithms.

5. Conclusion

- The energy consumption forecasting algorithms we deployed at the floor and building levels are performing admirably.
- The goals of the study, which were to examine the forecasts for the CU-BEMS office building's daily energy usage, have been accomplished.
- Using several models of prediction, we calculated the RSME, MSE, MAE, and accuracy. We were able to get accurate data on an hourly and daily basis, which is useful for predicting the building's future energy needs.

6. Future scope

Here are some potential future scopes and benefits of using these models:

- Methods for the coordinated management of air conditioners to lower peak demand.
- Building Efficiency and Sustainability Can Be Improved Through the Investigation and Development of New Energy Management Solutions, Potentially Made Possible by the Use of ARIMA, LSTM, and XG Boost Models.
- Anomaly detection of sensors and AC that are not functioning properly

7. References

7.1 Information sources

[1]. Pipattanasomporn, M., Chitalia, G., Songsiri, J. et al. CU-BEMS, smart building electricity consumption and indoor environmental sensor datasets. Sci Data 7, 241 (2020). https://doi.org/10.1038/s41597-020-00582-3

[2]. ExxonMobil. 2018 Outlook for Energy: A View to 2040. Available online: https://corporate.exxonmobil.com/en/~/media/Global/Files/outlook-for-energy/2018-Outlook-for-Energy.pdf. (accessed on 25 March 2019).

[3]. Alduailij, M.A., Petri, I., Rana, O. et al. Forecasting peak energy demand for smart buildings. J Supercomput 77, 6356–6380 (2021). https://doi.org/10.1007/s11227-020-03540-3