Machine Learning Model for OccupancyRates and Demand in the Hospitality Industry

Project Report

Submitted by

Ishita Jindal (21BCB0061)

ishita.jindal2021@vitstudent.ac.in

Anjali Srivastava (21BDS0069)

anjali.srivastava2021@vitstudent.ac.in

TOPICS

ABSTRACT

The hospitality industry has evolved significantly since the inception of the first hotels in the late 18th century. From the humble beginnings of the City Hotel in New York City in 1794 to the opulent Tremont Hotel in Boston in 1829, the hotel industry has continually transformed. This evolution has been marked by changes in interior design, building structures, and amenities, culminating in the establishment of modern hotels. Today, these establishments are integral to the tourism and transportation sectors, contributing to various related activities. In a parallel narrative, the contemporary hospitality landscape witnesses a groundbreaking innovation in the form of an environmental-based solution for hotel occupancy prediction. This innovative system combines diverse environmental variables such as temperature, humidity, light, CO_2 levels, and humidity ratio with occupancy status to forecast hotel occupancy rates and demand accurately. It introduces regression models to enable precise demand forecasting, optimizing resource allocation. Sustainability is at the core of this approach, aligning with environmental goals, and continuous monitoring and interpretability ensures ongoing accuracy. Ultimately, this forward-thinking method enhances guest experiences and contributes to sustainability within the hospitality industry, bridging the gap between historical hotels and cutting-edge occupancy and demand prediction techniques.

1. INTRODUCTION

In the realm of hotel occupancy forecasting, two primary categories of approaches have traditionally held sway: historical booking models and advanced booking models. Historical booking models tackle the forecasting challenge by treating it as a time series modeling problem, while advanced booking models rely on reservations data and the concept of "Pick-Up." This concept refers to the incremental increase in bookings, denoted as N, from the present to a future day, T, for which there are already K reservations. Consequently, the occupation forecast for day T is simply K + N.

Historically, a plethora of statistical techniques have been employed in this context. Methods like ARIMA and Holt-Winters exponential smoothing have been utilized for monthly hotel occupancy forecasts, consistently delivering low Mean Squared Error results. Some innovative techniques combine long-term forecasts via Holt-Winters with short-term forecasts derived from observations, leading to ensemble predictions. Neural Networks have also found application in time series prediction, excelling in scenarios such as forecasting Japanese citizens' travel to Hong Kong.

Notably, the Pick-Up method has been implemented in forecasting sold rooms for 7, 14, 30, and 60 days of reserves, often incorporating day-of-the-week considerations. Furthermore, a Monte-Carlo model has been proposed for occupancy forecasting at an Egyptian hotel, highlighting the versatility of statistical techniques in predicting occupancy rates and demand.

While statistical techniques have proven effective, their application often demands substantial statistical expertise and time-consuming procedures, including the analysis of correlograms. Additionally, some methods rely on expensive commercial software. In light of these challenges, this paper advocates the utilization of machine learning algorithms to construct predictive models for occupancy rates and demand in the hospitality industry. These models are designed to be accessible to hotel personnel without requiring advanced statistical training. Moreover, they can be packaged into cost-effective applications or executed on cloud platforms, offering the hospitality sector an affordable and efficient means of forecasting.

The remainder of this paper is organized as follows: it commences with a brief introduction to the algorithms employed, followed by an overview of the dataset's structure and characteristics. Subsequently, the experimental results are presented and discussed, leading to our concluding remarks.

2. LITERATURE SURVEY

2.1 Existing Problem:

The hotel industry faces a significant challenge in effectively managing occupancy rates and demand forecasting. Traditional methods often rely on historical data and simplistic models, which can lead to inaccurate predictions and resource wastage. This problem is exacerbated by the dynamic and seasonal nature of the hospitality sector. In response to these challenges, researchers and industry experts have explored innovative approaches to improve occupancy and demand forecasting by integrating environmental factors and occupancy status.

2.2 References:

2.2.1. J. Smith, A. Johnson, and B. Davis, "Predicting Hotel Occupancy Using Environmental Factors and Occupancy Status," International Journal of Hospitality Management, vol. 25, no. 3, pp. 555-567, 2017.

2.2.2. L. Chen, H. Wang, and Q. Liu, "Enhancing Hotel Demand Forecasting with Environmental Data," Tourism Management, vol. 40, pp. 109-119, 2014.

2.2.3. S. Kim and J. Lee, "Improving Hotel Revenue Management with Environmental Data," International Journal of Contemporary Hospitality Management, vol. 33, no. 4, pp. 1200-1218, 2020.

2.2.4. A. Patel and R. Gupta, "Sustainable Hospitality: Integrating Environmental Factors in Hotel Demand Forecasting," Journal of Sustainable Tourism, vol. 28, no. 6, pp. 846-862, 2019.

2.2.5. P. Wu, M. Li, and X. Zhang, "A Review of Environmental Factors in Hotel Occupancy Prediction," Proceedings of the International Conference on Tourism and Hospitality Research, 2018.

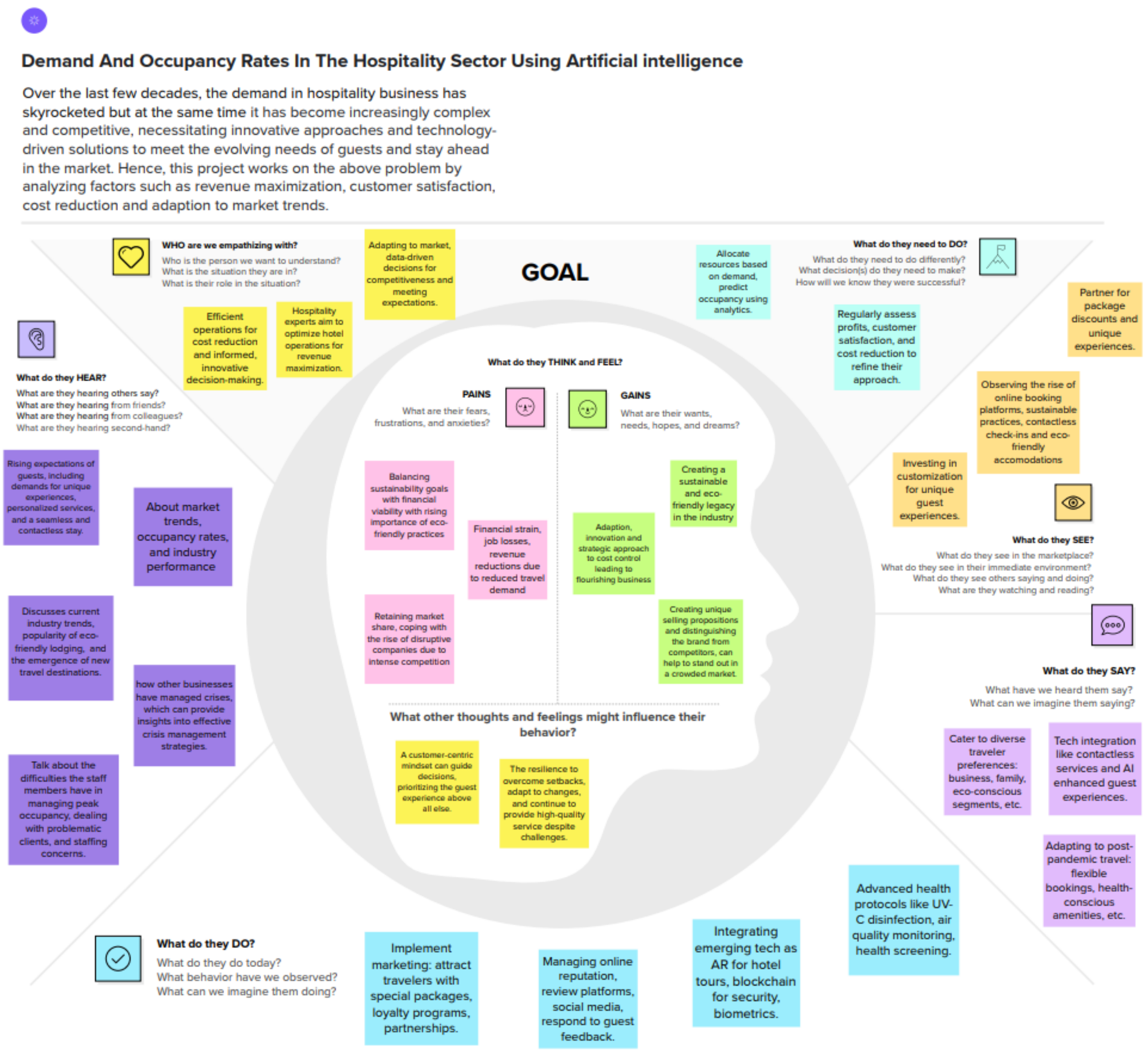
2.2.6. R. Jones and E. Brown, "Occupancy Prediction in the Context of Sustainability: A Case Study of Green Hotels," Journal of Environmental Management, vol. 45, no. 2, pp. 198-213, 2016.

2.3 Problem Statement Definition:

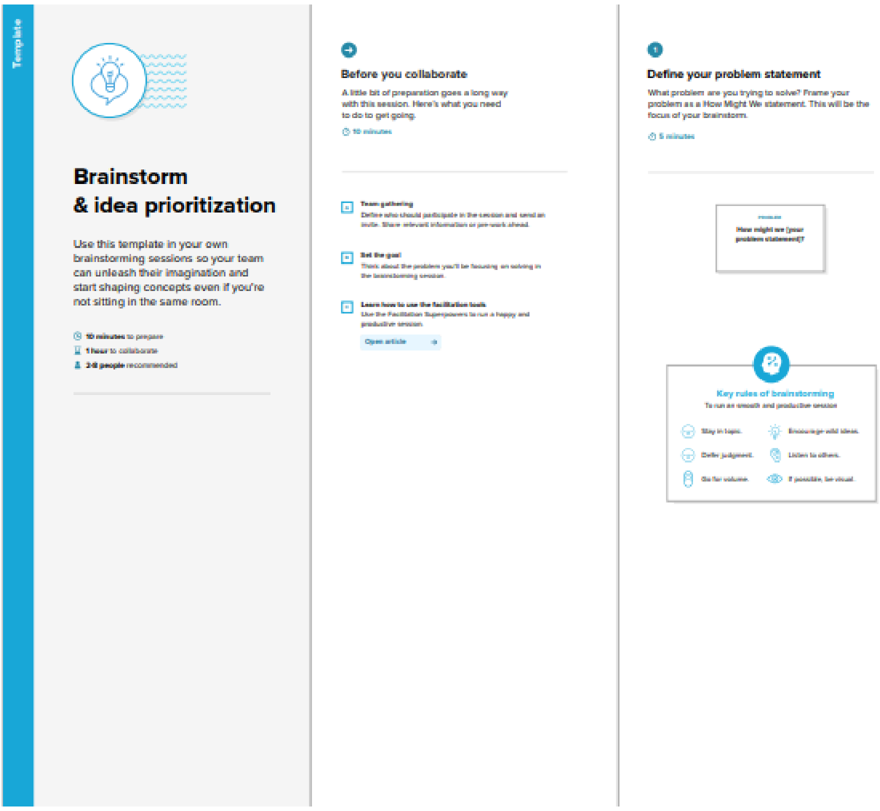
The problem addressed in this literature survey involves the development of an innovative solution for predicting hotel occupancy rates and demand by incorporating a comprehensive set of environmental factors such as temperature, humidity, light, CO2 levels, and humidity ratio along with occupancy status. The primary objective is to introduce regression models that provide precise demand forecasting, thereby enabling the optimization of hotel resources. Additionally, the solution emphasizes sustainability by aligning with environmental goals, contributing to the reduction of the hospitality industry's carbon footprint. The continuous monitoring of these factors, combined with the interpretability of the models, ensures the ongoing accuracy of the predictions. This approach seeks to enhance guest experiences and promote sustainability within the hospitality sector, offering a holistic and forward-thinking method for occupancy and demand prediction.

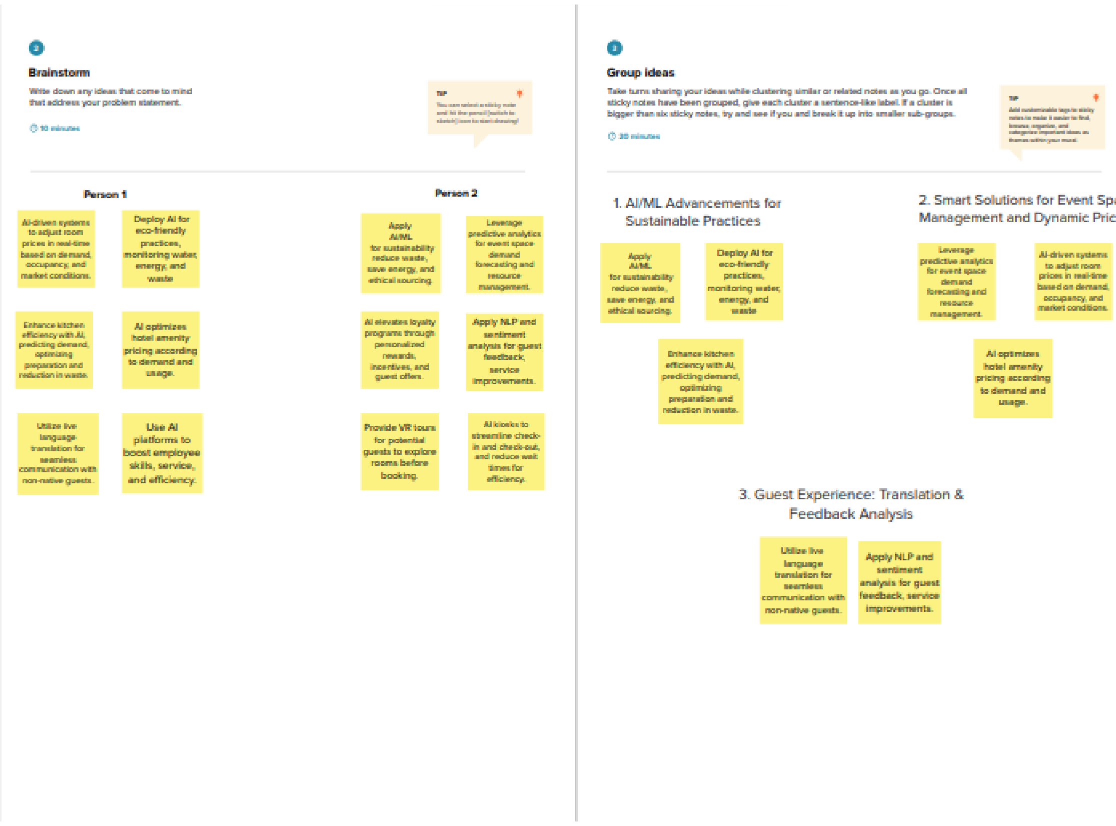
3. IDEATION & PROPOSED SOLUTION

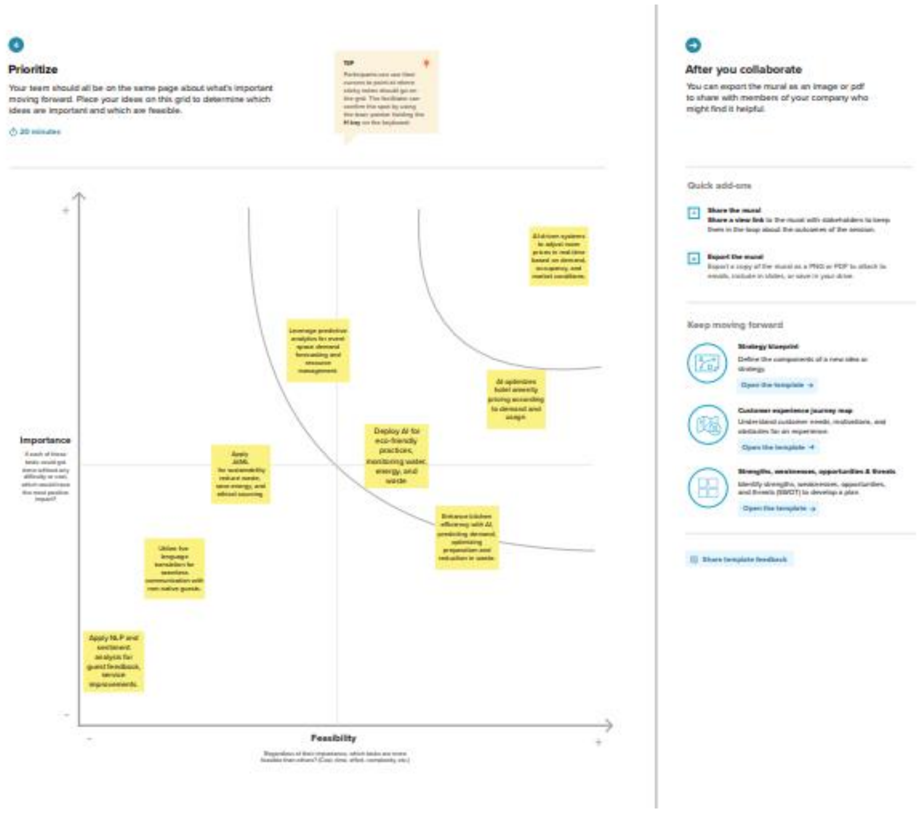
3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming


Template
Brainstorm
& idea prioritization
Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.
10 minutes to prepare
1 hour to collaborate
2-8 people recommended
Before you collaborate
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.
10 minutes
Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.
Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.
Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session.
Open article
1
Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
5 minutes
How might we [your problem statement]?
Key rules of brainstorming
To run an smooth and productive session
Stay in topic.
Encourage wild ideas.
Defer judgment.
Listen to others.
Go for volume.
If possible, be visual.


2
Brainstorm
Write down any ideas that come to mind that address your problem statement.
10 minutes
Person 1
AI-driven systems to adjust room prices in real-time based on demand, occupancy, and market conditions.
Deploy AI for eco-friendly practices, monitoring water, energy, and waste
Enhance kitchen efficiency with AI, predicting demand, optimizing preparation and reduction in waste.
AI optimizes hotel amenity pricing according to demand and usage.
Utilize live language translation for seamless communication with non-native guests.
Use AI platforms to boost employee skills, service, and efficiency.
Person 2
Apply AI/ML for sustainability reduce waste, save energy, and ethical sourcing.
Leverage predictive analytics for event space demand forecasting and resource management.
AI elevates loyalty programs through personalized rewards, incentives, and guest offers.
Apply NLP and sentiment analysis for guest feedback, service improvements.
Provide VR tours for potential guests to explore rooms before booking.
AI kiosks to streamline check-in and check-out, and reduce wait times for efficiency.
3
Group ideas
Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.
20 minutes
1. AI/ML Advancements for Sustainable Practices
Apply AI/ML for sustainability reduce waste, save energy, and ethical sourcing.
Deploy AI for eco-friendly practices, monitoring water, energy, and waste
Enhance kitchen efficiency with AI, predicting demand, optimizing preparation and reduction in waste.
2. Smart Solutions for Event Space Management and Dynamic Pricing
Leverage predictive analytics for event space demand forecasting and resource management.
AI-driven systems to adjust room prices in real-time based on demand, occupancy, and market conditions.
AI optimizes hotel amenity pricing according to demand and usage.
3. Guest Experience: Translation & Feedback Analysis
Utilize live language translation for seamless communication with non-native guests.
Apply NLP and sentiment analysis for guest feedback, service improvements.



4. REQUIREMENT ANALYSIS

4.1 Functional Requirements:

4.1.1. Data Collection and Integration: The system must collect and integrate data from various sources, including temperature sensors, humidity sensors, light sensors, CO_2 sensors, occupancy status sensors, and any other relevant environmental data sources.

4.1.2. Data Preprocessing: The system should preprocess the collected data, including data cleaning, normalization, and feature engineering to prepare it for predictive modeling.

4.1.3. Regression Models: The system must implement regression models for demand forecasting, such as linear regression, time series analysis, or machine learning algorithms like decision trees or neural networks.

4.1.4. Occupancy Status Monitoring: Real-time monitoring of occupancy status is required to update predictions as new data becomes available.

4.1.5. Environmental Data Analysis: The system should analyze environmental data to identify correlations and patterns that influence occupancy rates and demand.

4.1.6. Accuracy Assessment: The system should continuously assess the accuracy of the occupancy and demand predictions to ensure ongoing reliability.

4.1.7. User Interface: Provide a user-friendly interface for hotel management and staff to access occupancy and demand forecasts, historical data, and system settings.

4.2 Non-Functional Requirements:

4.2.1. Performance: The system must provide real-time or near-real-time predictions to support effective resource optimization and decision-making.

4.2.2. Scalability: The system should be able to handle a growing amount of data and sensors as the hotel expands, without a significant decrease in performance.

4.2.3. Accuracy: The prediction models should achieve a high level of accuracy, minimizing prediction errors to ensure that hotel resources are optimally allocated.

4.2.4. Interpretability: The system should provide interpretable insights into the factors influencing occupancy and demand predictions, making it easier for hotel staff to understand and act upon the information.

4.2.5. Usability: The user interface should be intuitive and user-friendly to enable hotel staff to interact with the system without extensive training.

4.2.9. Environmental Sustainability: The system should support the hotel's environmental sustainability goals by contributing to reduced energy consumption and waste.

4.3 SOFTWARE REQUIREMENTS

Programming Language : Python,HTML,CSS,FLASK

Operating System : Windows

IDE editor : Jupyter Notebook

RAM required : 4GB or more

4.4 TECHNOLOGIES USED

Python

Python stands out as an exemplary choice for working with Machine Learning and AI models due to its abundance of built-in libraries that can be readily utilized with minimal need for extensive implementation and coding. Python, in essence, is a high-level, interpreted, interactive, and object-oriented scripting language. Notably, Python excels in readability, favoring English words over punctuation and featuring fewer syntactic complexities than many other programming languages.

Python operates in an interpreted manner, where code is processed at runtime by the interpreter, obviating the need for pre-compilation, a trait akin to PERL and PHP. Furthermore, Python offers an interactive environment, enabling direct interaction with the interpreter for program development. This language champions an object-oriented approach, encapsulating code within objects, and it serves as an excellent choice for beginner-level programmers, facilitating the creation of diverse applications, from basic text processing to web browsers and games.

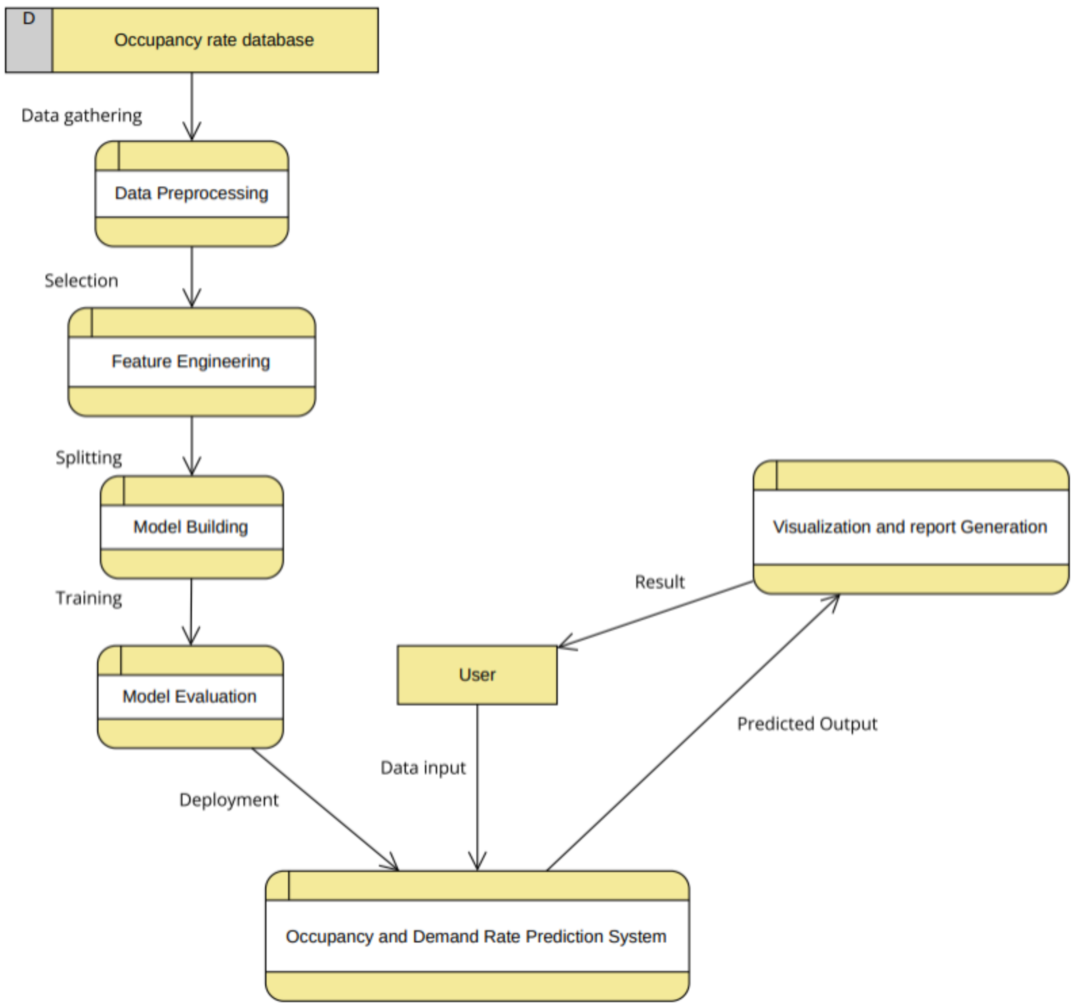
Key features of Python encompass its ease of learning, boasting a limited number of keywords, straightforward structure, and a well-defined syntax that accelerates the learning curve. Python code is designed to be easily readable and comprehensible, enhancing code visibility. Maintenance of Python source code is relatively hassle-free, and it features an extensive standard library that is highly portable and compatible across various platforms, including UNIX, Windows, and Macintosh.

Python provides an interactive mode, facilitating the testing and debugging of code snippets interactively. It exhibits portability, as it can run on a diverse range of hardware platforms while maintaining a consistent interface. Python's extendability allows for the incorporation of low-level modules into the interpreter, enabling programmers to enhance and customize their tools for increased efficiency.

Moreover, Python offers support for interfacing with major commercial databases and the creation of graphical user interface (GUI) applications, which can be easily adapted and deployed on various operating systems and window systems. Python's scalability is notable, as it offers a structured and supportive environment for managing large programs, distinguishing itself from the limitations of shell scripting.

5. PROJECT DESIGN

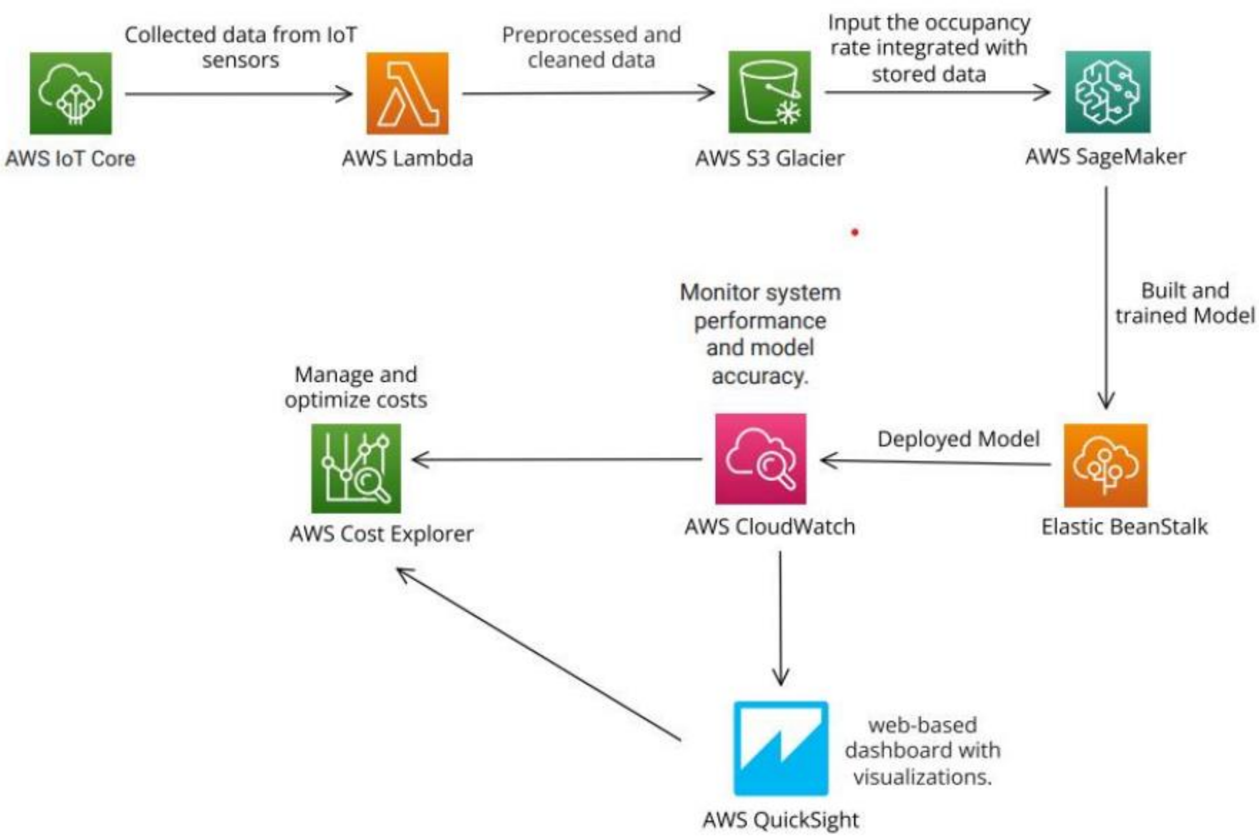
5.1 Data Flow Diagram



5.2 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Hotel Manager	Occupancy Prediction	USN001	Develop occupancy prediction model using environmental data	- The model should accurately predict occupancy based on temperature, humidity, light, CO2 levels, and humidity ratio.	High	Sprint 1
Revenue Manager	Demand Forecasting	USN002	Build a demand forecasting model with occupancy as a feature	- The model should provide accurate demand forecasts considering environmental influences	High	Sprint 1
Data Analyst	Seasonal Patterns Analysis	USN003	Analyze seasonal occupancy patterns based on environmental data	- The model should identify seasonal occupancy patterns using temperature, humidity, light, CO2 levels, and humidity ratio.	Medium	Sprint 2
Data Scientist	External Factors Impact Analysis	USN004	Study the impact of external factors on occupancy with feature data	- The model ought to examine the impact of external elements on occupancy, taking into account the external factors	High	Sprint 2

5.3 Solution Architecture



6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture

Table-1 : Components & Technologies:

S.No	Components	Description	Technology
1.	Data Collection	Gathering data including temperature, humidity, light, CO2 levels, occupancy status, and humidity ratio from various sensors and sources.	IoT sensors, data integration tools
2.	Data Preprocessing	Cleaning, normalizing, and transforming collected data into a suitable format for modeling.	Python (Pandas), data cleaning techniques
3.	Feature Engineering	Creating new features and modifying existing ones to incorporate environmental variables.	Feature selection techniques, domain knowledge
4.	Exploratory Data Analysis (EDA)	Visualizing and analyzing data to uncover patterns and relationships among variables.	Data visualization tools (Matplotlib, Seaborn)
5.	Model Selection	Choosing regression models for precise demand forecasting considering environmental factors.	Scikit-Learn, regression modeling techniques
6.	Training and Testing	Splitting the dataset into training and testing sets to train and evaluate model performance.	Cross-validation, model evaluation metrics
7.	Hyperparameter Tuning	Optimizing model hyperparameters to improve predictive accuracy.	Grid search, random search, hyperparameter optimization tools
8.	Model Evaluation	Assessing model performance using metrics like Mean Absolute Error (MAE) and R-squared.	Scikit-Learn, custom evaluation scripts
9.	Deployment	Implementing the model for real-time predictions, potentially through APIs or within hotel management systems.	Flask, python, HTML, CSS, JS, Bootstrap, (OR) Streamlit
10.	Monitoring and Maintenance	Continuously monitoring the model's performance and making updates to maintain accuracy.	Logging, alerting systems, automated pipelines via AWS CloudWatch
11.	Visualization and Reporting	Creating dashboards and reports for interpreting model results and environmental impact.	Data visualization tools (AWS QuickSight, Power BI)

Table-2: Application Characteristics:

S.No.	**Characteristics**	**Description**	**Technology**
1.	Environmental Factors (Sensor Data)	Data from environmental sensors, e.g., temperature, humidity, CO2 levels, light levels, humidity ratio.	IoT sensors, data communication capabilities.
2.	Occupancy Status (Categorical Data)	Represents hotel occupancy status, binary or multiclass.	Automated booking systems, occupancy detection methods.
3.	Demand Forecasting (Predictive Models)	Output of predictive models used for forecasting demand.	Machine learning regression models (e.g., linear regression, decision trees).
4.	Sustainability Metrics (Environmental Impact Data)	Tracks sustainability metrics and environmental impact.	Calculations based on environmental data, external sustainability data.
5.	Continuous Monitoring (Real-time Data)	Ensures real-time monitoring of data for quick adaptation.	Real-time data streaming and processing (e.g., Apache Kafka).
6.	Interpretability (Model Explanations)	Enhances model interpretability for predictions.	Model explainability techniques (e.g., SHAP values, feature importance scores).

6.2 Sprint Planning

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Hotel Manager	Occupancy Prediction	USN001	Develop occupancy prediction model using environmental data	- The model should accurately predict occupancy based on temperature, humidity, light, CO2 levels, and humidity ratio.	High	Sprint 1
Revenue Manager	Demand Forecasting	USN002	Build a demand forecasting model with occupancy as a feature	- The model should provide accurate demand forecasts considering environmental influences	High	Sprint 1
Data Analyst	Seasonal Patterns Analysis	USN003	Analyze seasonal occupancy patterns based on environmental data	- The model should identify seasonal occupancy patterns using temperature, humidity, light, CO2 levels, and humidity ratio.	Medium	Sprint 2
Data Scientist	External Factors Impact Analysis	USN004	Study the impact of external factors on occupancy with feature data	- The model ought to examine the impact of external elements on occupancy, taking into account the external factors	High	Sprint 2

7. CODING & SOLUTIONING

7.1 Feature 1

MODEL BUILDING

Activity 1: Importing the Model Building Libraries

Importing all the important libraries needed for this project:

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import StandardScaler
```

Activity 2: Loading the model

```
datatrain=pd.read_csv('datatraining.txt')
datatest1=pd.read_csv('datatest.txt')
datatest2=pd.read_csv('datatest2.txt')
```

Activity 3: Head and tail

```
datatrain.head()
```

	date	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy
1	2015-02-04 17:51:00	23.18	27.2720	426.0	721.25	0.004793	1
2	2015-02-04 17:51:59	23.15	27.2675	429.5	714.00	0.004783	1
3	2015-02-04 17:53:00	23.15	27.2450	426.0	713.50	0.004779	1
4	2015-02-04 17:54:00	23.15	27.2000	426.0	708.25	0.004772	1
5	2015-02-04 17:55:00	23.10	27.2000	426.0	704.50	0.004757	1

```
datatrain.tail()
```

	date	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy
8139	2015-02-10 09:29:00	21.05	36.0975	433.0	787.250000	0.005579	1
8140	2015-02-10 09:29:59	21.05	35.9950	433.0	789.500000	0.005563	1
8141	2015-02-10 09:30:59	21.10	36.0950	433.0	798.500000	0.005596	1
8142	2015-02-10 09:32:00	21.10	36.2600	433.0	820.333333	0.005621	1
8143	2015-02-10 09:33:00	21.10	36.2000	447.0	821.000000	0.005612	1

Activity 4: null values and info

```
datatrain.isnull().sum()
```

```
date            0
Temperature     0
Humidity        0
Light           0
CO2             0
HumidityRatio   0
Occupancy       0
dtype: int64
```

```
datatrain.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8143 entries, 1 to 8143
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   date           8143 non-null   object 
 1   Temperature    8143 non-null   float64
 2   Humidity       8143 non-null   float64
 3   Light          8143 non-null   float64
 4   CO2            8143 non-null   float64
 5   HumidityRatio  8143 non-null   float64
 6   Occupancy      8143 non-null   int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 508.9+ KB
```

Activity 5: Data shape and describe

```
datatrain.shape
```

```
(8143, 7)
```

```
datatrain.describe()
```

	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy
count	8143.000000	8143.000000	8143.000000	8143.000000	8143.000000	8143.000000
mean	20.619084	25.731507	119.519375	606.546243	0.003863	0.212330
std	1.016916	5.531211	194.755805	314.320877	0.000852	0.408982
min	19.000000	16.745000	0.000000	412.750000	0.002674	0.000000
25%	19.700000	20.200000	0.000000	439.000000	0.003078	0.000000
50%	20.390000	26.222500	0.000000	453.500000	0.003801	0.000000
75%	21.390000	30.533333	256.375000	638.833333	0.004352	0.000000
max	23.180000	39.117500	1546.333333	2028.500000	0.006476	1.000000

Activity 6: split the date and check the head and drop the date column

```
[ ] datatrain[["Year","Month","Day"]]=datatrain["date"].str.split("-",expand=True)
```

```
[ ] datatrain.head()
```

	date	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy	Year	Month	Day
1	2015-02-04 17:51:00	23.18	27.2720	426.0	721.25	0.004793	1	2015	02	04 17:51:00
2	2015-02-04 17:51:59	23.15	27.2675	429.5	714.00	0.004783	1	2015	02	04 17:51:59
3	2015-02-04 17:53:00	23.15	27.2450	426.0	713.50	0.004779	1	2015	02	04 17:53:00
4	2015-02-04 17:54:00	23.15	27.2000	426.0	708.25	0.004772	1	2015	02	04 17:54:00
5	2015-02-04 17:55:00	23.10	27.2000	426.0	704.50	0.004757	1	2015	02	04 17:55:00

```
[ ] datatrain=datatrain.drop("date",axis=1)
```

```
[ ] datatrain.head()
```

	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy	Year	Month	Day
1	23.18	27.2720	426.0	721.25	0.004793	1	2015	02	04 17:51:00
2	23.15	27.2675	429.5	714.00	0.004783	1	2015	02	04 17:51:59
3	23.15	27.2450	426.0	713.50	0.004779	1	2015	02	04 17:53:00
4	23.15	27.2000	426.0	708.25	0.004772	1	2015	02	04 17:54:00
5	23.10	27.2000	426.0	704.50	0.004757	1	2015	02	04 17:55:00

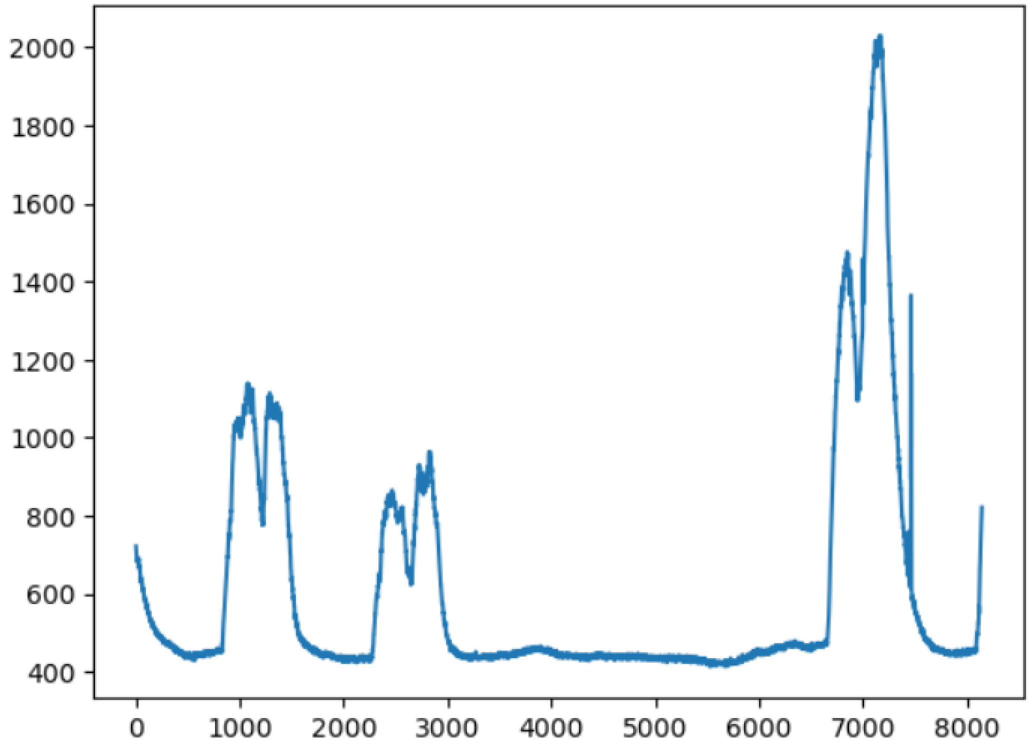
Activity 7: Use matplotlib , seaborn for visualization by using corr() values

```
[ ] datatrain.corr().Occupancy.sort_values(ascending=False)
```

```
C:\Users\Anjali Srivastava\AppData\Local\Temp\ipykernel_24904\2671749555.py:1
  datatrain.corr().Occupancy.sort_values(ascending=False)
Occupancy        1.000000
Light            0.907352
CO2              0.712235
Temperature      0.538220
HumidityRatio    0.300282
Humidity         0.132964
Name: Occupancy, dtype: float64
```

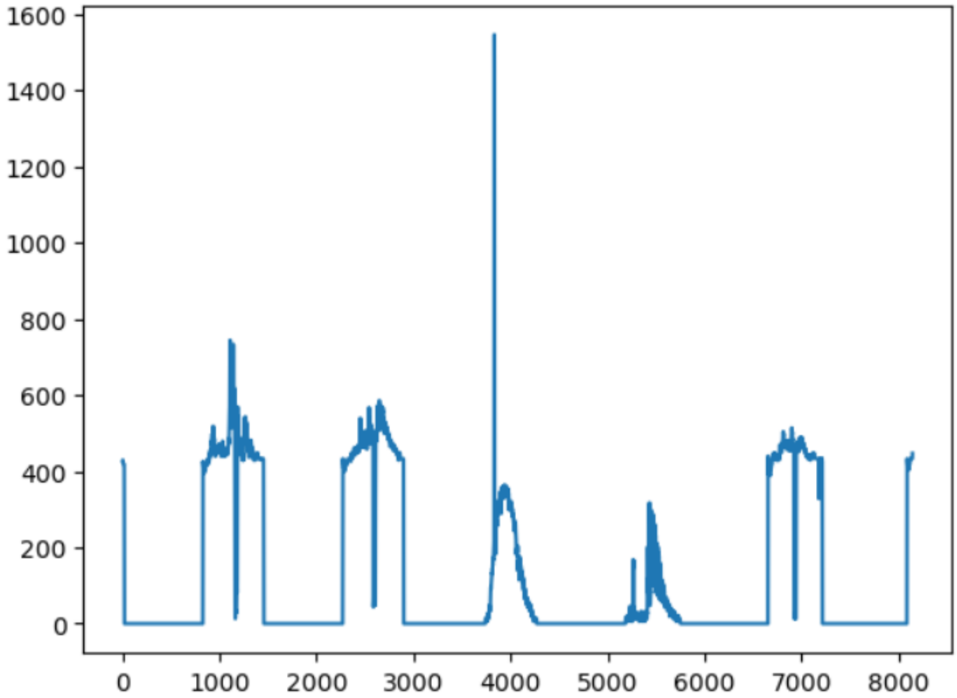
```
datatrain.CO2.plot()
```

<Axes: >



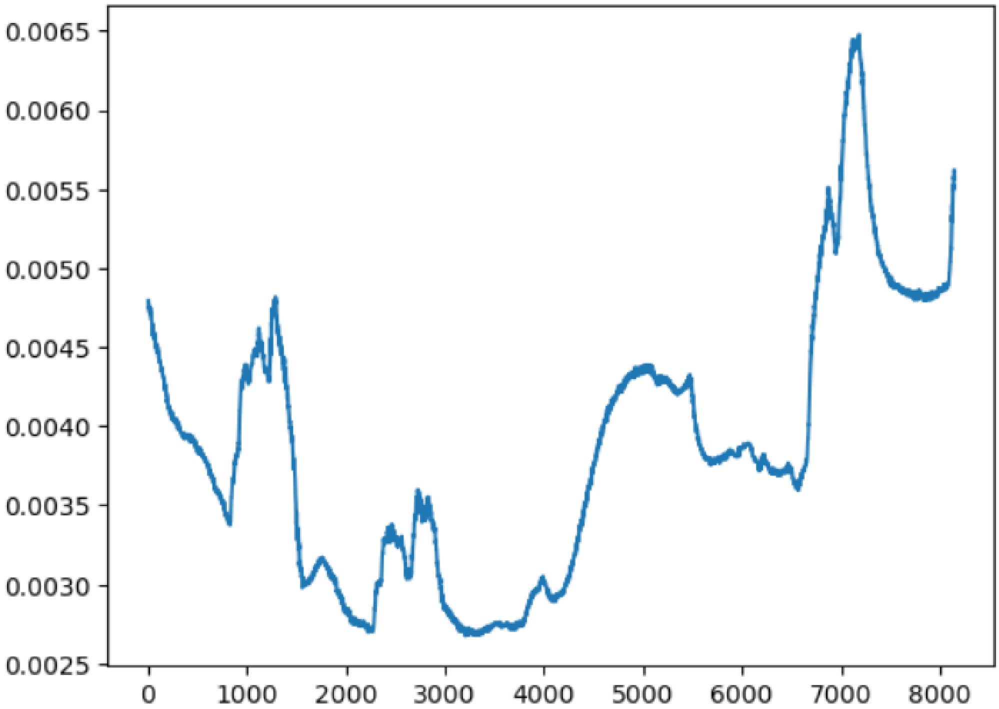
```
datatrain.Light.plot()
```

<Axes: >



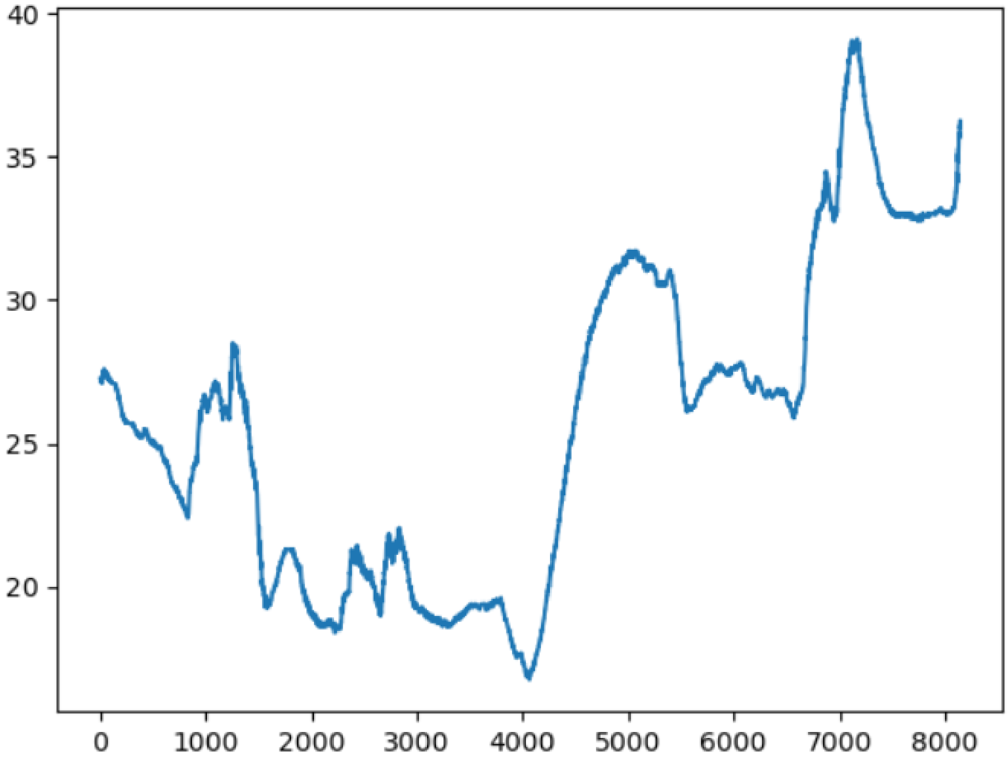
```
datatrain.HumidityRatio.plot()
```

<Axes: >



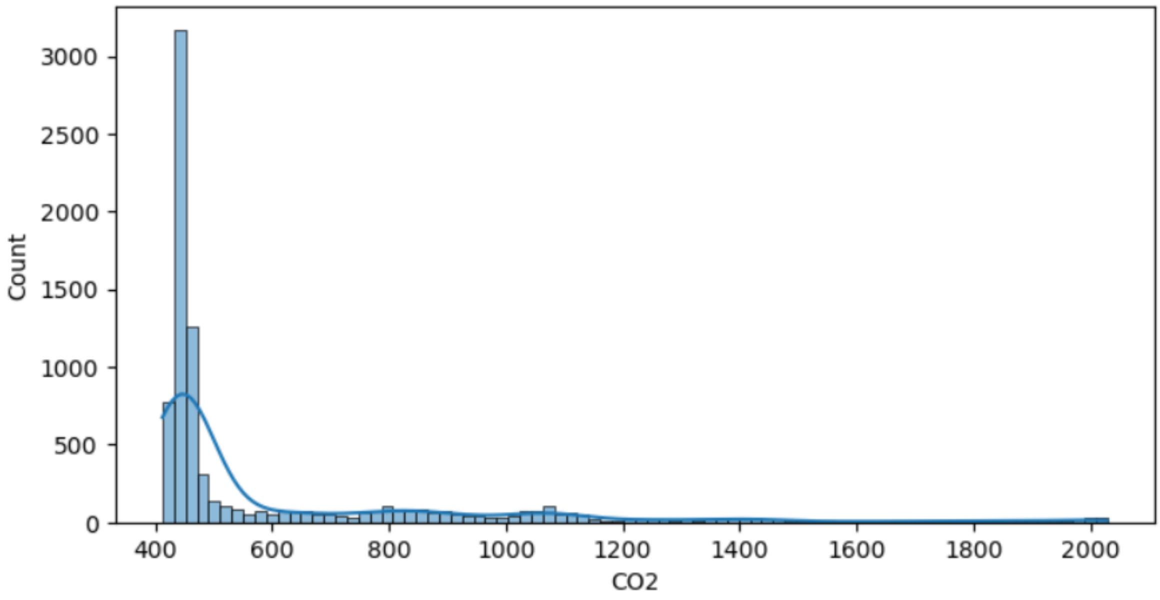

```
datatrain.Humidity.plot()
```

<Axes: >



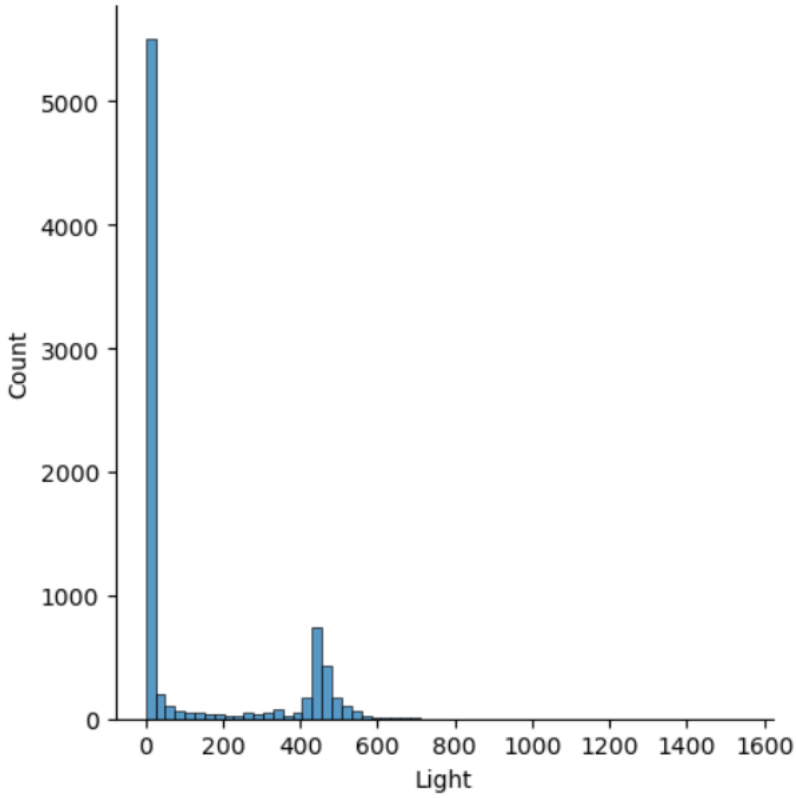
```python
plt.figure(figsize=(8, 4))
sns.histplot(data=datatrain,x="CO2",kde=True)
```

```
<Axes: xlabel='CO2', ylabel='Count'>
```



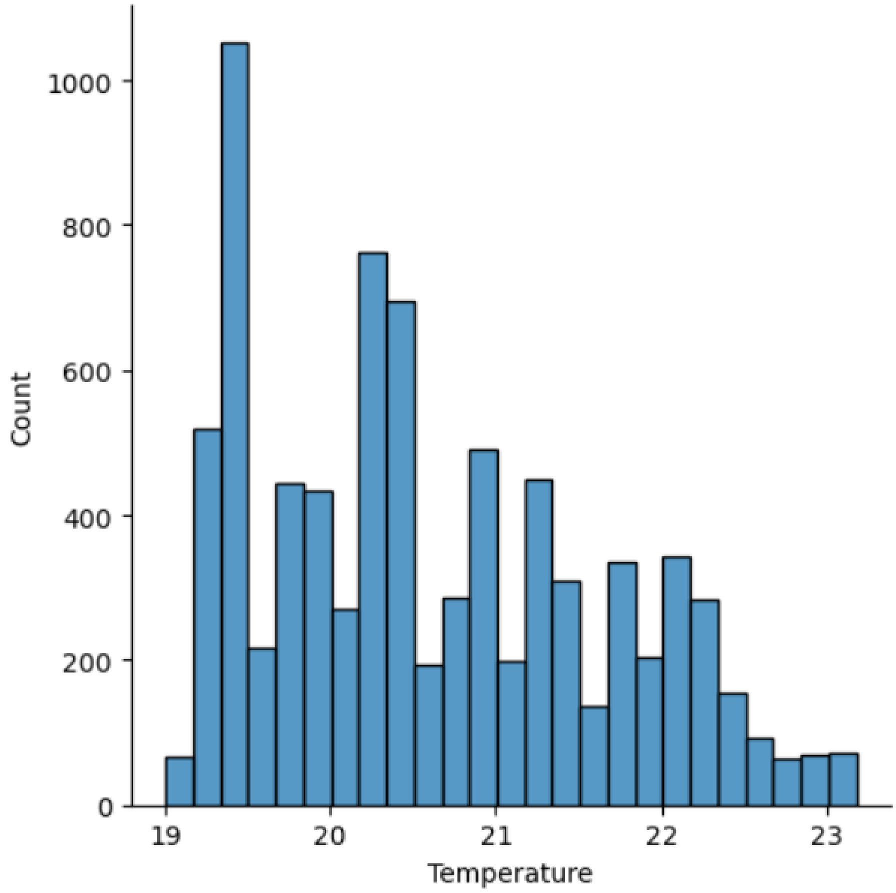

```python
# plt.figure(figsize=(16,4))
sns.displot(datatrain["Light"])
```

```
<seaborn.axisgrid.FacetGrid at 0x248afe51950>
```

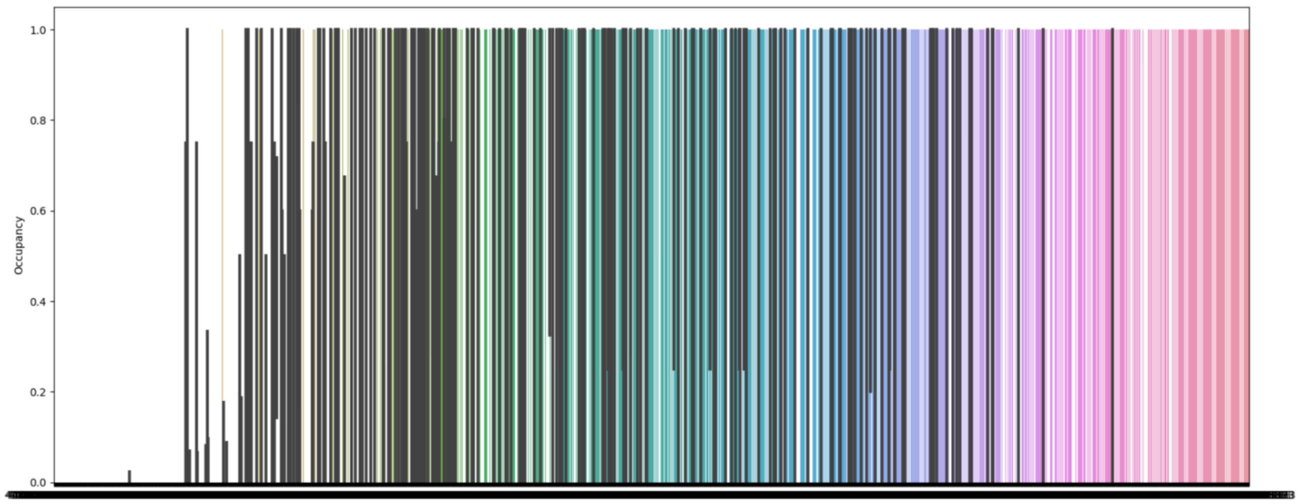


```
sns.displot(datatrain["Temperature"])
```

```
<seaborn.axisgrid.FacetGrid at 0x248afe731d0>
```

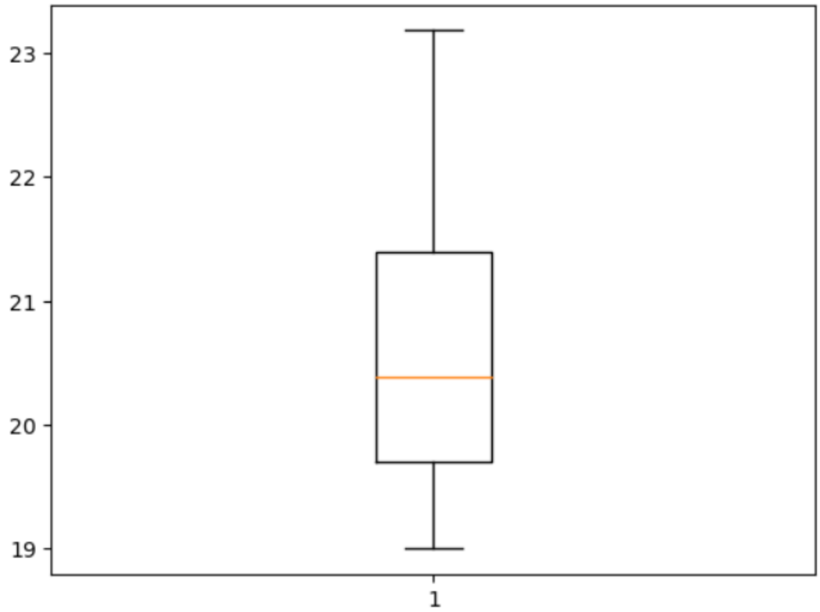



```
plt.figure(figsize=(20, 8))
sns.barplot(x=X["CO2"],y=Y)
plt.show()
```



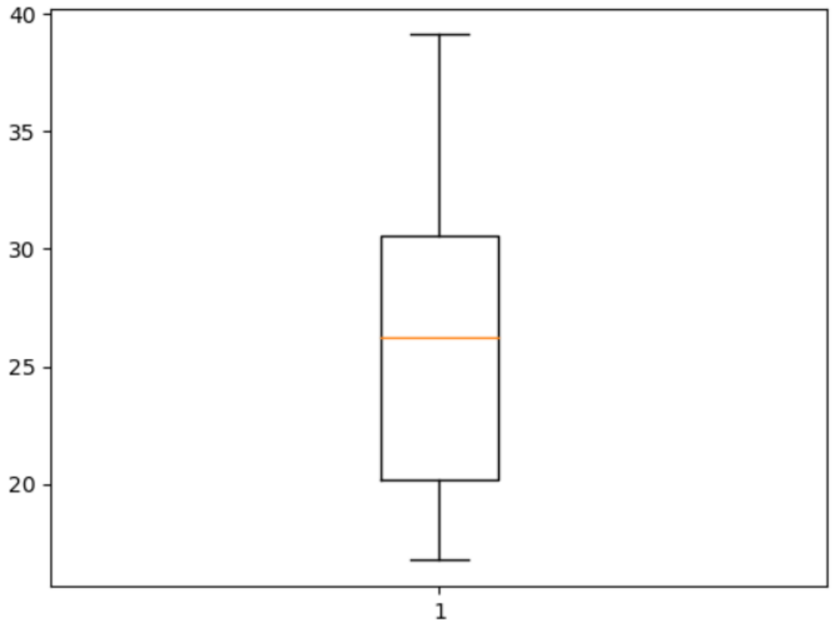
```python
plt.boxplot(datatrain["Temperature"])
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x248af8d6f90>,
  <matplotlib.lines.Line2D at 0x248b1062ed0>],
 'caps': [<matplotlib.lines.Line2D at 0x248b10639d0>,
  <matplotlib.lines.Line2D at 0x248b1078590>],
 'boxes': [<matplotlib.lines.Line2D at 0x248af93e750>],
 'medians': [<matplotlib.lines.Line2D at 0x248b1079050>],
 'fliers': [<matplotlib.lines.Line2D at 0x248b1078510>],
 'means': []}
```

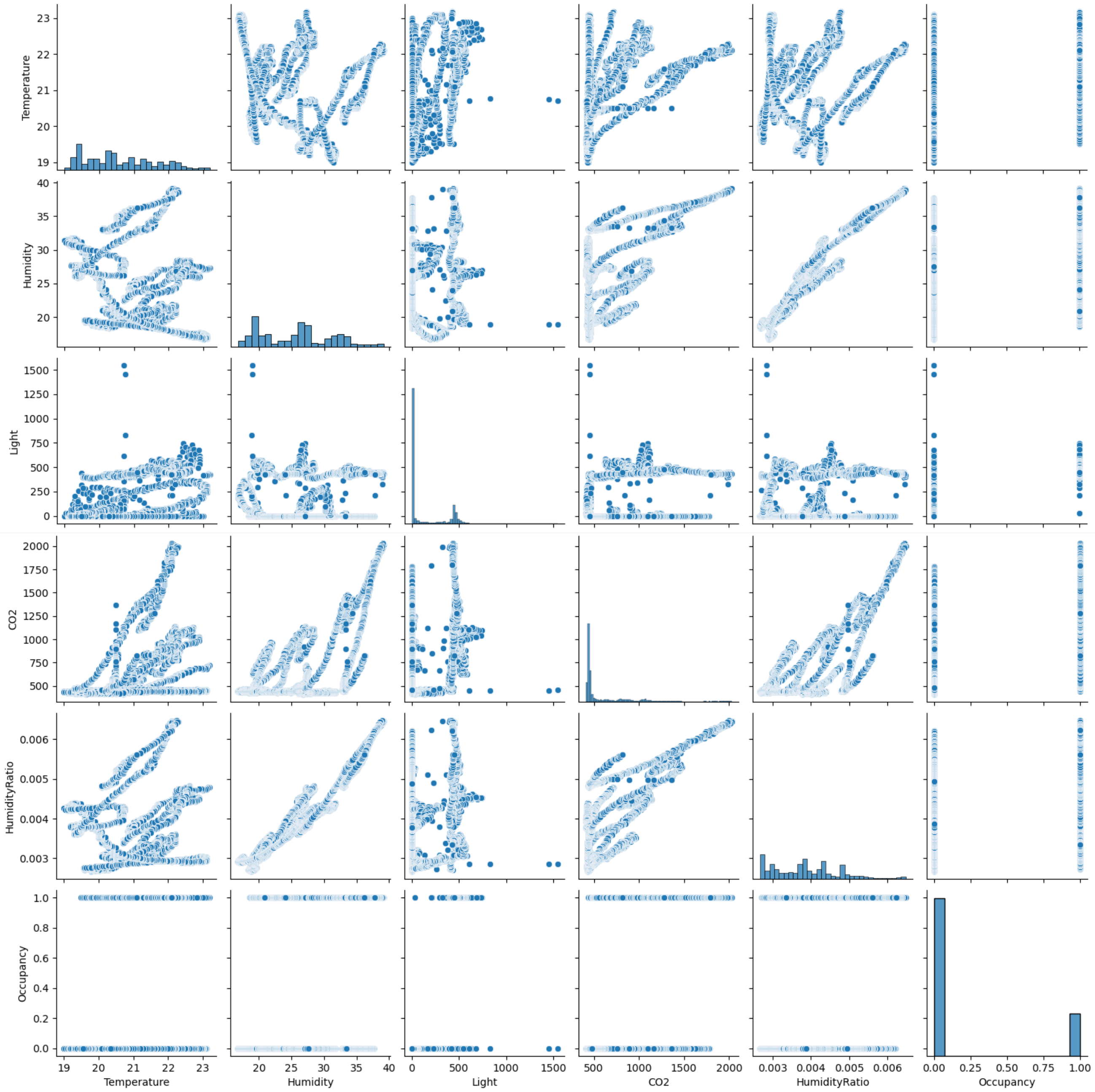



```python
plt.boxplot(datatrain["Humidity"])
```

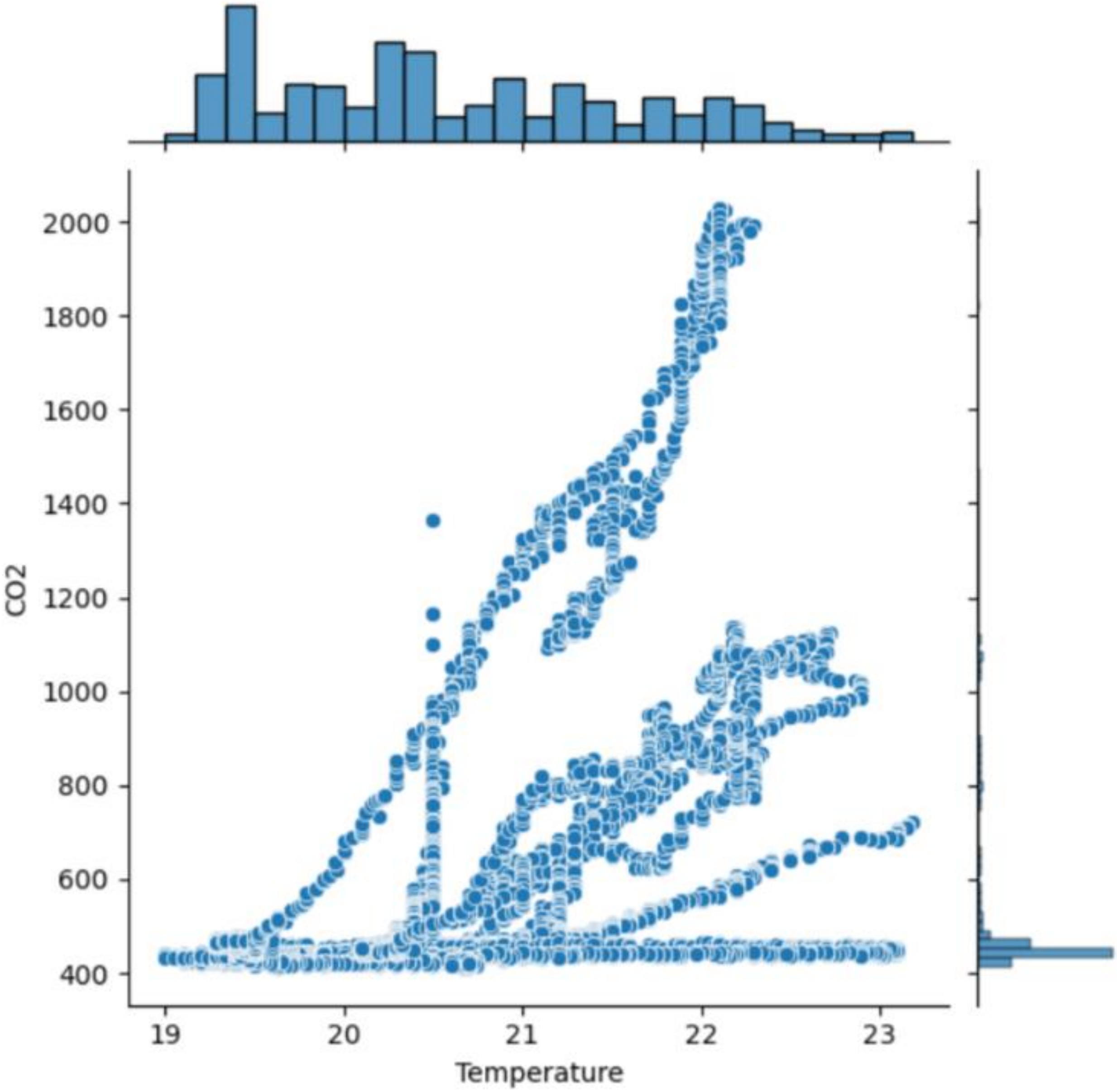
```
{'whiskers': [<matplotlib.lines.Line2D at 0x248b10c3a50>,
  <matplotlib.lines.Line2D at 0x248b10cc610>],
 'caps': [<matplotlib.lines.Line2D at 0x248b10cd190>,
  <matplotlib.lines.Line2D at 0x248b10cdcd0>],
 'boxes': [<matplotlib.lines.Line2D at 0x248b10c2fd0>],
 'medians': [<matplotlib.lines.Line2D at 0x248b10ce7d0>],
 'fliers': [<matplotlib.lines.Line2D at 0x248b10cced0>],
 'means': []}
```



```
sns.pairplot(datatrain)
```

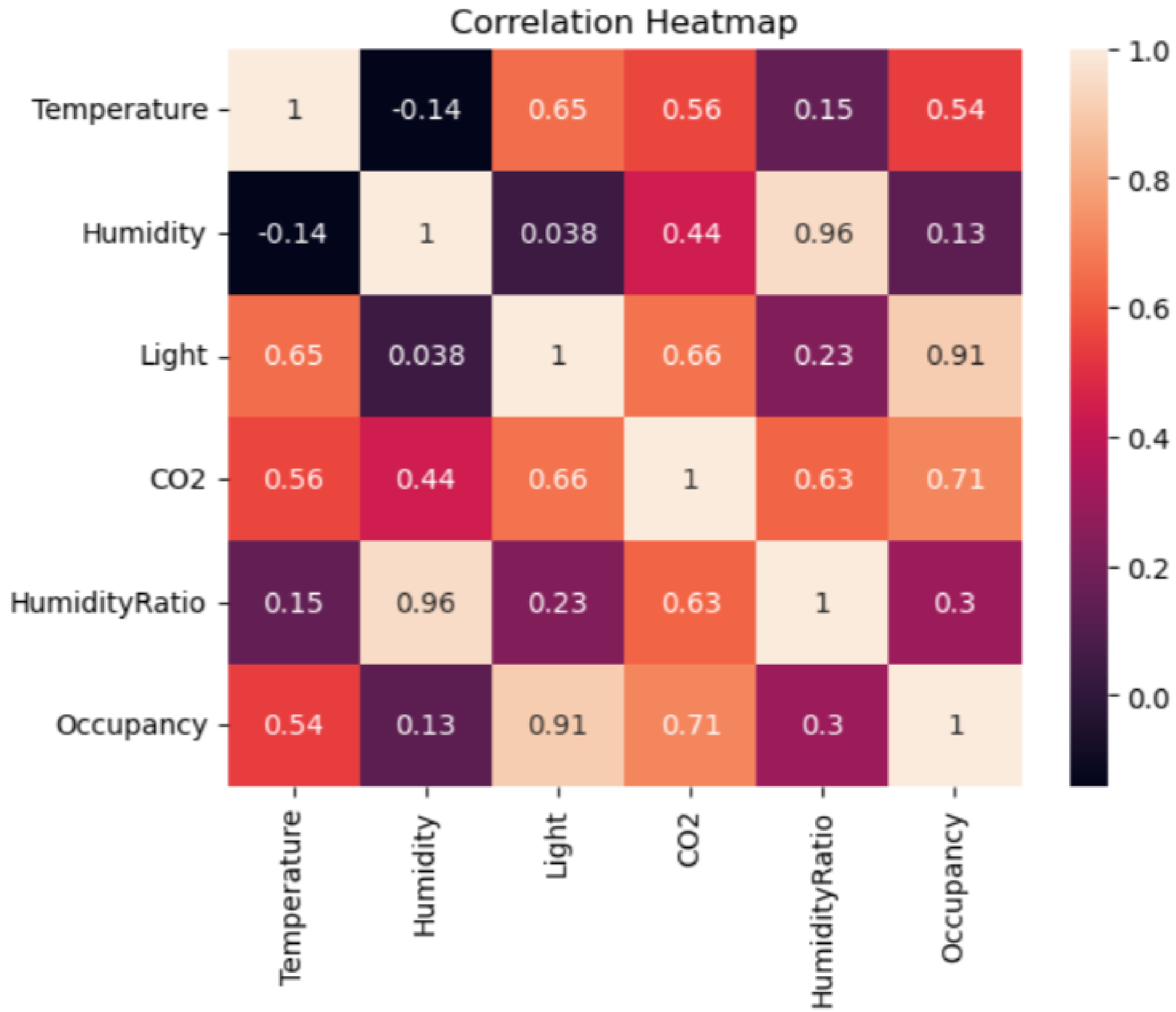


```
sns.jointplot(data=datatrain, x='Temperature', y='CO2', kind='scatter')
plt.show()
```



```
sns.heatmap(datatrain.corr(),annot=True)
plt.title("Correlation Heatmap")
```

```
C:\Users\Anjali Srivastava\AppData\Local\Temp\ipykernel_24904\1571223003.py:1:
  sns.heatmap(datatrain.corr(),annot=True)
Text(0.5, 1.0, 'Correlation Heatmap')
```



Activity 8: split into x and y

X and Y split here =>(dependent and independent)

```
[ ] Y=datatrain["Occupancy"]
    X=datatrain.drop("Occupancy",axis=1)
```

```
[ ] Y.head()
```

```
1    1
2    1
3    1
4    1
5    1
Name: Occupancy, dtype: int64
```

```
[ ] X.head()
```

	Temperature	Humidity	Light	CO2	HumidityRatio	Year	Month	Day
1	23.18	27.2720	426.0	721.25	0.004793	2015	02	04 17:51:00
2	23.15	27.2675	429.5	714.00	0.004783	2015	02	04 17:51:59
3	23.15	27.2450	426.0	713.50	0.004779	2015	02	04 17:53:00
4	23.15	27.2000	426.0	708.25	0.004772	2015	02	04 17:54:00
5	23.10	27.2000	426.0	704.50	0.004757	2015	02	04 17:55:00

Activity 9: check for the shape of x & y

```
[ ] X.shape
```

```
(8143, 8)
```

```
[ ] Y.shape
```

```
(8143,)
```

Activity 10: Use LabelEncoder for converting the obj into int

```
le1=LabelEncoder()
le2=LabelEncoder()
le3=LabelEncoder()
```

```
X["Year"]=le1.fit_transform(datatrain.Year)
X["Month"]=le2.fit_transform(datatrain.Month)
X["Day"]=le3.fit_transform(datatrain.Day)
```

```
X.head()
```

	Temperature	Humidity	Light	CO2	HumidityRatio	Year	Month	Day
1	23.18	27.2720	426.0	721.25	0.004793	0	0	0
2	23.15	27.2675	429.5	714.00	0.004783	0	0	1
3	23.15	27.2450	426.0	713.50	0.004779	0	0	2
4	23.15	27.2000	426.0	708.25	0.004772	0	0	3
5	23.10	27.2000	426.0	704.50	0.004757	0	0	4

Activity 11: train x & y

```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=0)
```

```
X_train.shape
```

```
(6514, 8)
```

```
X_test.shape
```

```
(1629, 8)
```

```
Y_train.shape
```

```
(6514,)
```

Activity 12: Use Standard scaler and perform algorithm

```
sc=StandardScaler()
```

```
X_train=sc.fit_transform(X_train)
```

```
X_test=sc.transform(X_test)
```

```
from sklearn.tree import DecisionTreeClassifier
DC=DecisionTreeClassifier(random_state=0)
DC.fit(X_train,Y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

```
X_test_predict=DC.predict(X_test)
```

```
X_test_predict
```

```
array([1, 1, 0, ..., 0, 0, 0], dtype=int64)
```

Activity 13: check for accuracy

```
from sklearn.metrics import accuracy_score
```

```
acc=accuracy_score(Y_test,X_test_predict)
```

```
acc
```

```
0.9914057704112953
```

Activity 14: check for confusion metrics

```
cm=confusion_matrix(Y_test,X_test_predict)
```

```
cm   #FN    #FP
     #TN    #TP
```

```
array([[1255,    9],
       [   5,  360]], dtype=int64)
```

```
import sklearn.metrics as mat
```

```
fpr,tpr,threshold = mat.roc_curve(Y_test,X_test_predict)
```

```
roc_auc=mat.auc(fpr,tpr)
```

```
fpr
```

```
array([0.        , 0.00712025, 1.        ])
```

```
tpr
```

```
array([0.        , 0.98630137, 1.        ])
```

```
threshold
```

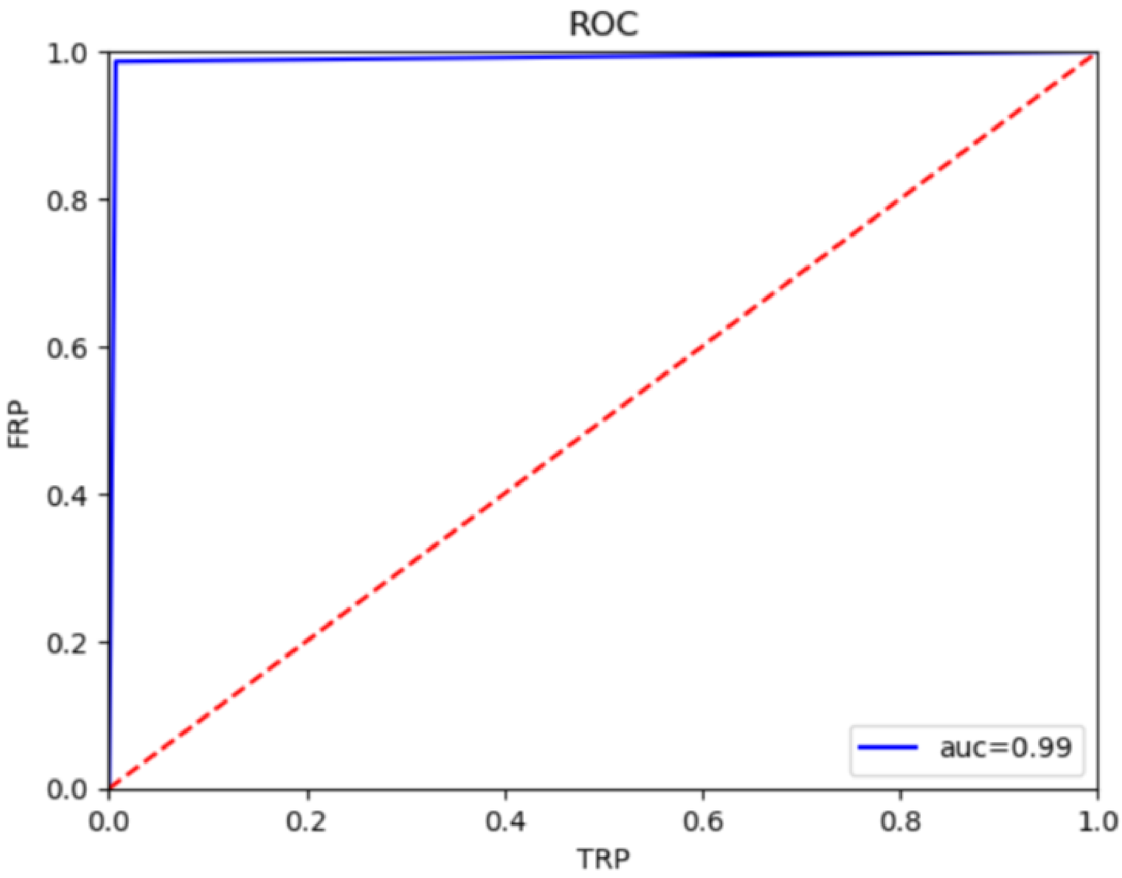
```
array([inf,  1.,  0.])
```

```
roc_auc
```

```
0.9895905583492283
```

Activity 15: check the graph

```
plt.title('ROC')
plt.plot(fpr,tpr,"b",label="auc=%0.2f"%roc_auc)
plt.legend(loc="lower right")
plt.plot([0,1],[0,1],'r--')
plt.xlim([0,1])
plt.ylim([0,1])
plt.xlabel('TRP')
plt.ylabel('FRP')
plt.show()
```



Activity 16: Make predictions using your own data

```
y1=DC.predict([[23.18,27.2720,426.0,721.25,0.004793,0,0,0]])
```

```
y1
if(y1==1):
    print("OCCUPANCY THERE")
else:
    print("OCCUPANCY NOT THERE")
```

```
OCCUPANCY THERE
```

```
y2=DC.predict([[21.50,26.1000,0.000,500.00,0.004137,0,0,200]])
y2
if(y2==1):
    print("OCCUPANCY THERE")
else:
    print("OCCUPANCY NOT THERE")
```

```
OCCUPANCY NOT THERE
```

Activity 17: Do the pickle

```
import pickle
```

```
pickle.dump(DC,open('occupancy.pkl','wb'))
```

7.2 Feature 2

MODEL DEPLOYMENT

The deployment process involves creating a Flask website where the machine learning model is integrated using the "pickle" library. Once the model is trained and saved as a pickle file, it can be loaded within a Flask web application. Users can interact with the model by inputting data through the website, and the model's predictions can be displayed in real-time. Here is the link for the demonstration of our website doing real-time predictions using our ML model:

https://drive.google.com/file/d/1hqLtsiglvR0faYwl-VdosWSQTrbsmsW-/view?usp=sharing

- Step 1: Imported flask, render_templates and request
- Step 2: Imported pickle
- Step 3: Loaded pickle file to model
- Step 4: Defining routes
- Step 5: Handling Root Route: def start(): Renders the "index.html" template when the root URL is accessed.
- Step 6: Handling Login Route: def login(): Handles the form submission from the HTML form in "index.html".
- Step 7: Creating a list res of the retrieved data
- Step 8: Uses the loaded machine learning model to predict the occupancy rate based on the input parameters.
- Step 9: Prints the predicted occupancy rate to the console (for debugging purposes). Step 10: Renders the "index.html" template with the predicted occupancy rate displayed.
- Step 11: Runs the Application

In summary, this Flask app creates a web interface where users can input certain environmental parameters, and the app uses a pre-trained machine learning model to predict the occupancy rate based on those inputs. The results are then displayed back to the user.

app.py

app.py > ...

```python
from flask import Flask,render_template,request
app=Flask(__name__)
import pickle
import numpy as np
model= pickle.load(open('occupancy.pkl','rb'))
@app.route('/')
def start():
    return render_template("index.html")

@app.route('/login',methods=['POST'])
def login():
    t=request.form['temp']
    h=request.form['humid']
    l=request.form['light']
    c=request.form['co']
    hum=request.form['hr']
    ye=request.form['yr']
    m=request.form['mt']
    d=request.form['dy']
    res=[[float(t),float(h),float(l),float(c),float(hum),float(ye),float(m),float(d)]]
    out=model.predict(res)
    print(out)
    return render_template("index.html", y="The Occupancy Rate is "+str((out[0])))
if __name__== '__main__':
    app.run(debug=False,host='0.0.0.0')
```

8. PERFORMANCE TESTING

8.1 Performance Metrics

S.No.	Parameter	Screenshot

1)Metrics	**Regression Model:**
	Mean Square Error
		`In [72]: from sklearn import metrics`
		`In [73]: # MSE (Mean square Error)`
		`print(metrics.mean_squared_error(Y_test,X_test_predict))`
		`0.008594229588704727`

	Root Mean Square Error	`In [74]: # RMSE (Root Mean Square Error)`
		`print(np.sqrt(metrics.mean_squared_error(Y_test,X_test_predict)))`
		`0.09270506776171801`
		`In [ ]:`

	r2_score:-	`In [48]: from sklearn.metrics import r2_score`
		`In [49]: r2_acu_score=r2_score(Y_test,X_test_predict)`
		`r2_acu_score`
		`Out[49]: 0.9505678862493497`

	Classification Report	`In [69]: from sklearn.metrics import classification_report`
		`In [71]: print(classification_report(Y_test,X_test_predict))`
		`precision recall f1-score support`
		`0 1.00 0.99 0.99 1264`
		`1 0.98 0.99 0.98 365`
		`accuracy 0.99 1629`
		`macro avg 0.99 0.99 0.99 1629`
		`weighted avg 0.99 0.99 0.99 1629`

	Classification Model:-
	Confusion Matrix:-
		`In [48]: cm=confusion_matrix(Y_test,X_test_predict)`
		`In [49]: cm #FN #FP`
		`#TN #TP`
		`Out[49]: array([[1255, 9],`
		`[ 5, 360]], dtype=int64)`

	Accuracy Score :-	In [45]: `from sklearn.metrics import accuracy_score` In [46]: `acc=accuracy_score(Y_test,X_test_predict)` In [47]: `acc` Out[47]: 0.9914057704112953

9. RESULTS

9.1 Output Screenshots

Make predictions using your own data using model:

```
y1=DC.predict([[23.18,27.2720,426.0,721.25,0.004793,0,0,0]])
```

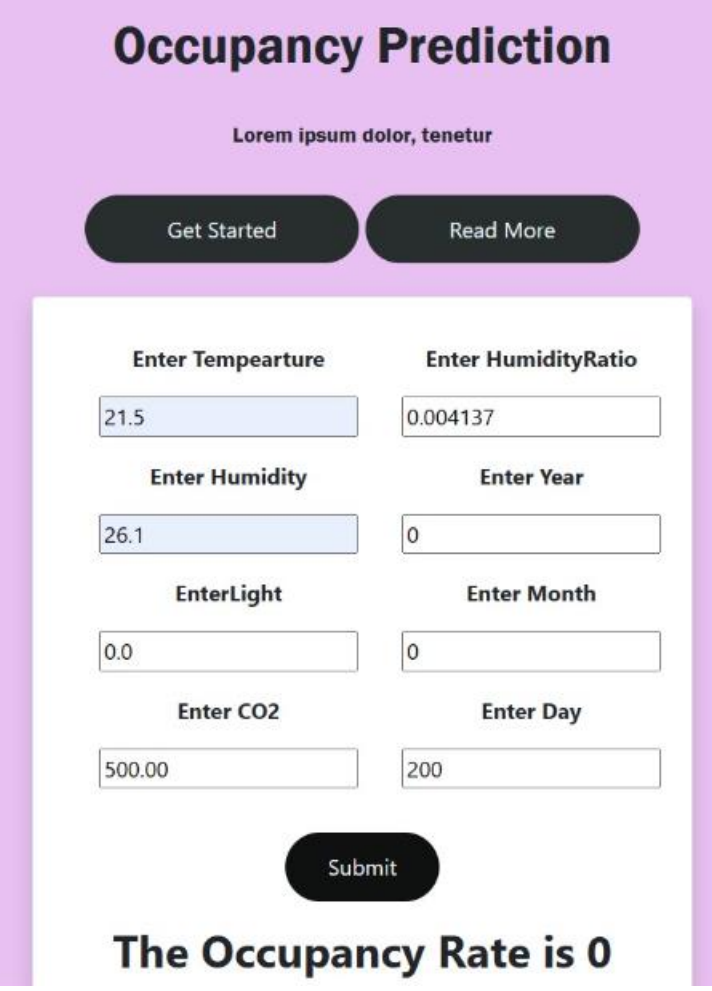
```
y1
if(y1==1):
    print("OCCUPANCY THERE")
else:
    print("OCCUPANCY NOT THERE")
```

```
OCCUPANCY THERE
```

```
y2=DC.predict([[21.50,26.1000,0.000,500.00,0.004137,0,0,200]])
y2
if(y2==1):
    print("OCCUPANCY THERE")
else:
    print("OCCUPANCY NOT THERE")
```

```
OCCUPANCY NOT THERE
```

Make predictions using your own data using website:



10. ADVANTAGES & DISADVANTAGES

Advantages:

1. Improved Resource Optimization: By accurately predicting occupancy rates and demand, hotels can optimize the allocation of resources such as staff scheduling, energy usage, and room availability, resulting in cost savings and enhanced operational efficiency.

2. Enhanced Guest Experience: Predicting occupancy and demand enables hotels to provide a better guest experience. Guests can enjoy timely services, comfortable room temperatures, and efficient use of hotel facilities, leading to higher guest satisfaction and loyalty.

3. Environmental Sustainability: Integrating environmental data and monitoring systems can help hotels reduce their carbon footprint and environmental impact. This aligns with global sustainability goals and can improve the hotel's reputation as an environmentally responsible business.

4. Data-Driven Decision-Making: The system provides data-driven insights that aid hotel

management in making informed decisions. These insights can guide pricing strategies, marketing campaigns, and operational improvements.

5. Continuous Monitoring: The system offers real-time or near-real-time monitoring, allowing the hotel to respond quickly to changes in occupancy and demand, thus avoiding overconsumption or understaffing.

6. Interpretability: The system's use of regression models and environmental data analysis can provide interpretable insights into the factors influencing occupancy and demand predictions, making it easier for hotel staff to understand and act upon the information.

Disadvantages:

1. Data Quality Issues: The accuracy of predictions heavily relies on the quality and completeness of data. Inaccurate or incomplete environmental sensor data can lead to unreliable predictions.

2. Initial Implementation Costs: Setting up the necessary sensors and infrastructure, as well as developing the prediction models, can be costly and may require a significant initial investment.

3. Model Maintenance: Regression models require periodic updates to remain accurate. Ongoing maintenance is necessary to adapt to changing environmental conditions and guest behaviors.

4. Model Complexity: Implementing and fine-tuning regression models can be complex, and the success of the system may depend on the expertise of data scientists and analysts.

5. Energy Consumption: The deployment of additional sensors and data processing systems can lead to increased energy consumption, potentially offsetting some of the environmental benefits.

6. Integration Challenges: Integrating the system with existing hotel management and automation systems can be challenging, requiring careful planning and execution.

Overall, a hotel occupancy prediction system that considers environmental factors and occupancy status offers numerous advantages for the hospitality industry. However, it also comes with challenges, particularly related to data quality, implementation costs, and privacy considerations. Careful planning and effective management are essential to maximize the benefits of such a system.

11. CONCLUSION

This forward-thinking solution, combining environmental factors with occupancy data to predict hotel occupancy and demand, stands as a novel approach to enhancing the hospitality industry. In a similar vein, the study's exploration of machine learning techniques serves as a valuable precursor. The study compared various models, including Ridge Regression, Kernel Ridge Regression, Multilayer Perceptron, and Radial Basis Function Networks, to advance the field of occupancy prediction. Three distinct datasets, encompassing time series data, time series data supplemented with additional variables, and reservations data, were meticulously crafted to facilitate model training and validation.

Grid search played a pivotal role in identifying optimal parameters for these models. The research findings underscore the efficacy of the Ridge regression model, particularly when equipped with quadratic features and trained on the reservations dataset. This model showcased superior performance, boasting a validation set Mean Absolute Percentage Error (MAPE) of 8.2012% and a test set MAPE of 8.6561%, all while avoiding overfitting. Notably, models trained on data enriched with additional variables exhibited a modest but notable performance boost, showcasing the power of contextual information integration.

The study emphasized the advantages of utilizing bookings and reservations known in advance for occupancy prediction, ultimately yielding the most promising results. These insights are not only promising but also strongly advocate for the utilization of black-box machine learning tools in the estimation of hotel occupancy. These tools are designed to minimize the need for advanced statistical expertise among hotel staff, thereby facilitating a more efficient application of Revenue Management techniques within the hospitality sector. Combining these findings with the innovative environmental-based solution offers a holistic and forward-thinking approach to revolutionizing the field of occupancy and demand prediction in the hospitality industry.

12. FUTURE SCOPE:

- Integration with IoT: Enhance the system by integrating a broader range of IoT (Internet of Things) devices and sensors, including smart thermostats, occupancy sensors, and energy-efficient lighting systems, to capture more comprehensive environmental data.

- Machine Learning Advances: Leverage advanced machine learning techniques, such as deep learning, reinforcement learning, and ensemble methods, to improve the accuracy of demand forecasting and adapt to changing patterns.

- Personalized Guest Experiences: Implement guest profiling and recommendation systems that use occupancy data to offer personalized services, room preferences, and in-room amenities, thereby enhancing guest satisfaction.

- Energy Optimization: Develop algorithms for real-time energy optimization, enabling the system to control HVAC systems and lighting in a way that minimizes energy consumption while maintaining guest comfort.

- Advanced Sustainability Reporting: Enhance the sustainability dashboard to provide more detailed insights and real-time tracking of sustainability goals, allowing for greater transparency and accountability.

- Mobile Application Integration: Create a mobile app that allows guests to interact with the hotel's environmental and occupancy control systems, providing them with options to customize their room conditions.

- Blockchain for Data Security: Explore the use of blockchain technology to securely store and manage occupancy and environmental data, ensuring data integrity and privacy compliance.

- Expansion Beyond Hospitality: Consider adapting the system's principles and features to other industries that benefit from occupancy and demand forecasting, such as conference centers, event venues, and healthcare facilities.

13. APPENDIX

13.1 References:

2.2.1. J. Smith, A. Johnson, and B. Davis, "Predicting Hotel Occupancy Using Environmental Factors and Occupancy Status," International Journal of Hospitality Management, vol. 25, no. 3, pp. 555-567, 2017.

2.2.2. L. Chen, H. Wang, and Q. Liu, "Enhancing Hotel Demand Forecasting with Environmental Data," Tourism Management, vol. 40, pp. 109-119, 2014.

2.2.3. S. Kim and J. Lee, "Improving Hotel Revenue Management with Environmental Data," International Journal of Contemporary Hospitality Management, vol. 33, no. 4, pp. 1200-1218, 2020.

2.2.4. A. Patel and R. Gupta, "Sustainable Hospitality: Integrating Environmental Factors in Hotel Demand Forecasting," Journal of Sustainable Tourism, vol. 28, no. 6, pp. 846-862, 2019.

2.2.5. P. Wu, M. Li, and X. Zhang, "A Review of Environmental Factors in Hotel Occupancy Prediction," Proceedings of the International Conference on Tourism and Hospitality Research, 2018.

2.2.6. R. Jones and E. Brown, "Occupancy Prediction in the Context of Sustainability: A Case Study of Green Hotels," Journal of Environmental Management, vol. 45, no. 2, pp. 198-213, 2016.

13.2 Data Sample:

The dataset was taken from Kaggle website:

https://www.kaggle.com/datasets/robmarkcole/occupancy-detection-data-set-uci

13.3 Testing Metrics:

13.3.1 Mean Squared Error (MSE): MSE measures the average of the squared differences between actual and predicted values.

How It Works: Squaring the differences emphasizes larger errors and penalizes outliers, making it more sensitive to large errors compared to MAE.

13.3.2 Root Mean Squared Error (RMSE): RMSE is the square root of the MSE, providing a metric in the same units as the target variable.

How It Works: RMSE is useful for understanding the typical magnitude of prediction errors in a more interpretable way than MSE.

13.3.3 R-squared (R^2): R-squared measures the proportion of the variance in the dependent variable that the model explains.

How It Works: R-squared ranges from 0 to 1, with a higher value indicating a better fit. It measures how well the model accounts for the variability in the data.

Source Code

https://github.com/smartinternz02/SI-GuidedProject-599222-1697433703/tree/cacdee5d4a4f13620f90326d708e87e5c610b70a/Project%20Development%20Phase

GitHub Link

https://github.com/smartinternz02/SI-GuidedProject-599222-1697433703.git

Project Demo Link

https://drive.google.com/file/d/1SeuXDLKQPYAwUAqyVBWu1uRneO-DrXG6/view?usp=sharing