SMART HOME TEMPERATURE PREDICTION USING MACHINE LEARNING

AN INDUSTRY ORIENTED MINI REPORT

Submitted to

JAWAHARLAL NEHRU TECNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING(AI&ML)

Submitted By

LIKHITHA PEGUDA	**21UK1A6683**
NIKHIL PARVATHAPU	**21UK1A6682**
LAXMIPRASANNA BOMMIDI	**21UK1A66C0**
SUSHMA BOMMIDI	**21UK1A66A9**

Under the guidance of

Ms. R. SRAVANI

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) 506005

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING(AI&ML)

VAAGDEVI ENGINEERING COLLEGE(WARANGAL)

CERTIFICATE OF COMPLETION
INDUSTRY ORIENTED MINI PROJECT

This is to certify that the UG Project Phase-1 entitled "SMART HOME TEMPERATURE PREDICTION USING MACHINE LEARNING" LIKHITHA PEGUDA(21UK1A6683),NIKHIL PARVATHAPU(21UK1A6682),LAXMIPRASANNA BOMMIDI(21UK1A66C0) SUSHMA GUNDA(21UK1A66A9) (in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2024- 2025.

Project Guide	**HOD**
Ms.R,SRAVANI	**Dr.K.SHARMILA**
(Assistant Professor)	(Professor)

EXTERNAL

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. SYED MUSTHAK AHMED,** Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this Mini Project in the institute.

We extend our heartfelt thanks to **Dr.K.SHARMILA**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the Mini Project.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the Mini Project and for their support in completing the Mini Project.

We express heartfelt thanks to the guide, **R.SRAVANI**, Assistant professor, Department of CSE for her constant support and giving necessary guidance for completion of this Mini Project.

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

LIKHITHA PEGUDA	**21UK1A6683**
NIKHIL PARVATHAPU	**21UK1A6682**
LAXMIPRASANNA BOMMIDI	**21UK1A66C0**
SUSHMA GUNDA	**21UK1A66A9**

ABSTRACT

Smart home temperature predicton involves forecasting indoor temperature based on sensor data and environment factors. Machine Learning algorithms play a crucial role in acheiving accurate predictions. Researchers combine the Internet of Things(IoT) WITH ML methods to optimize comfort and energy efficiency in smart buildings. Algorithms susch as Random Forest,Support Vector Machine,and Neural Networks have been used to predict indoor temperature. Additionally, incorporating future whether data improves prediction performance. These advancements contribute to creating energy-efficient and comfortable living spaces in smart homes.

TABLE OF CONTENTS:-

1.INTRODUCTION

1.1OVERVIEW

The user operates the smart home devices year in year out, have produced mass operation data, but these data do not be utilized well in past. Nowadays, these data can be used to predict user's behavior custom with the development of big data and machine learning technologies, and then the prediction results can be employed to enhance the intelligence of a smart home system.

In view of this, this paper proposes a novel unsupervised user behavior prediction (UUBP) algorithm, which employs an artificial neural network and proposes a forgetting factor to overcome the shortcomings of the previous prediction algorithm.

This algorithm has a high-level of autonomous and self-organizing learning ability while does not require too much human intervention. Furthermore, the algorithm can better avoid the influence of user's infrequent and out-of-date operation records, because of the forgetting factor.

Finally, the use of real end user's operation records to demonstrate that UUBP algorithm has a better level of performance than other algorithms from effectiveness.

1.2.PURPOSE

1.Energy Efficiency:

Accurate temperature predictions allow smart HVAC (heating, ventilation, and air conditioning) systems to optimize energy consumption.

By adjusting heating and cooling based on predicted temperatures, energy waste is minimized.

2.Comfort Optimization:

Predicting indoor temperatures ensures that occupants experience optimal comfort levels.

Smart thermostats can adjust settings in real-time to maintain desired temperatures.

3.Cost Savings:

Energy-efficient temperature control reduces utility bills.

Predictive models help homeowners make informed decisions about energy usage.

4.Health and Well-Being:

Stable indoor temperatures contribute to health and well-being.

Extreme temperature fluctuations can impact health, especially for vulnerable populations.

2.LITERATURE SURVEY

2.1EXISTING PROBLEM

Data Quality and Availability like obtaining accurate and reliable sensor data is crucial for ML models which include sensor errors, missing data, or noisy measurements can impact prediction accuracy.And Dynamic Environment like indoor temperature is influenced by various factors (occupancy, sunlight, appliances).ML models must adapt to changing conditions.

Overfitting includes complex ML models may overfit to historical data.Balancing model complexity and generalization is essential.

Across Homes like models trained on one home may not generalize well to others.Home-specific features (layout, insulation) affect predictions.Latency and Real-Time Prediction like smart homes require real-time predictions.Efficient algorithms and hardware are needed for low-latency responses. Generalization

Energy-Efficient Control includes ML models should optimize HVAC control for energy savings.Balancing comfort and energy efficiency is challenging.And user Behavior and Preferences like Predictions should align with user preferences.Learning individual preferences is complex. Smart home systems collect sensitive data.Ensuring data privacy and security is critical.

2.2 PROPOSED SOLLUTION

Feature Engineering Solution: Extract relevant features (time of day, occupancy patterns, weather) to improve prediction accuracy.

Ensemble Models Solution: Combine multiple ML models (e.g., Random Forest, Gradient Boosting) to enhance robustness and generalization.

Recurrent Neural Networks (RNNs) Solution: Use RNNs (e.g., LSTM) for time series data to capture temporal dependencies and predict future temperatures.

Adaptive Learning Solution: Continuously update models with new data to adapt to changing home dynamics.

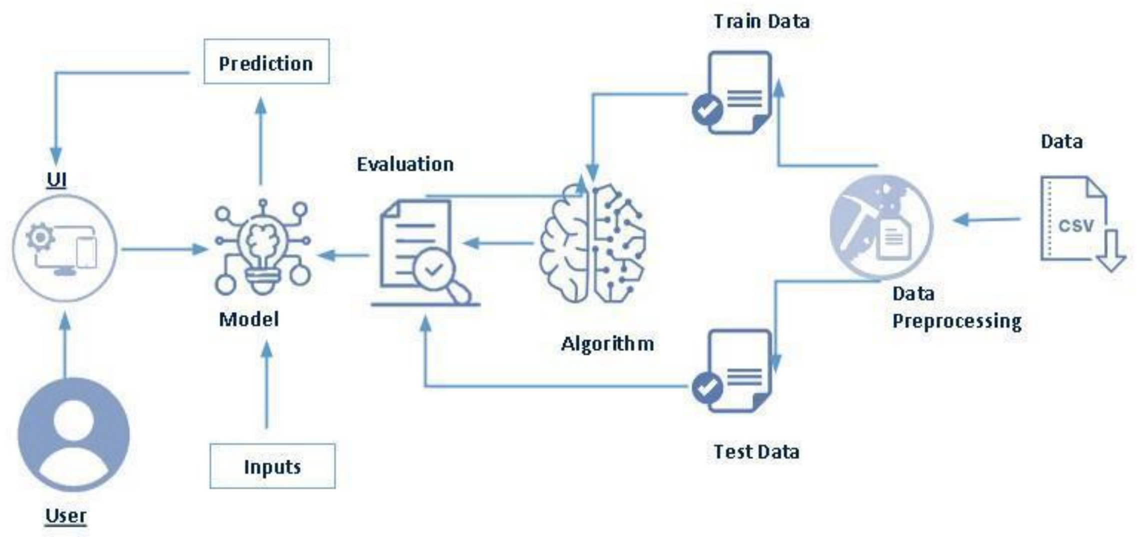
User Preferences Solution: Incorporate user preferences (comfort thresholds, energy-saving goals) into the prediction process.

Energy-Efficient Control Strategies Solution: Optimize HVAC control based on predicted temperatures to minimize energy consumption.

Interpretability Solution: Develop interpretable ML models to gain insights into temperature predictions.

3.THEORITICAL ANALYSIS

3.1. BLOCK DIAGRAM



3.2. SOFTWARE DESIGNING

The following is the Software required to complete this project:

➢ **Google Colab:** Google Colab will serve as the development and execution environment for your predictive modeling, data preprocessing, and model training tasks. It provides a cloud-based Jupyter Notebook environment with access to Python libraries and hardware acceleration.

➢ **Dataset (CSV File):** The dataset in CSV format is essential for training and testing your predictive model. It should include historical prediction data, Indoor temperature,Outdoor Temperature,Humidity,HVAC and other relevant features.

➢ **Data Preprocessing Tools:** Python libraries like NumPy, Pandas, and Scikit-learn will be used to preprocess the dataset. This includes handling missing data, feature scaling, and data cleaning.

Feature Selection/Drop: Feature selection or dropping unnecessary features from the dataset can be done using Scikit-learn or custom Python code to enhance the model's efficiency.

➢ **Model Training Tools:** Machine learning libraries such as Scikit-learn, TensorFlow, or PyTorch will be used to develop, train, and fine-tune the predictive model. Regression or classification models can be considered, depending on the nature of the temperature prediction task.

➢ **Model Accuracy Evaluation:** After model training, accuracy and performance evaluation tools, such as Scikit-learn metrics or custom validation scripts, will assess the model's predictive capabilities. You'll measure the model's ability to predict home temperature categories based on data.

➢ **UI Based on Flask Environment:** Flask, a Python web framework, will be used to develop the user interface (UI) for the system. The Flask application will provide a user-friendly platform for users to input location data or view temperature predictions, health information, and recommended precautions.

➢ Google Colab will be the central hub for model development and training, while Flask will facilitate user interaction and data presentation. The dataset, along with data preprocessing, will ensure the quality of the training data, and feature selection will optimize the model. Finally, model accuracy evaluation will confirm the system's predictive capabilities, allowing users to rely on the temperature predictions and associated health information.

EXPERIMENTAL INVESTIGATION

In this project, we have used Smart Home temperature prediction using machine learning Dataset. This dataset is a csv file consisting of labelled data and having the following columns-

1.CO2_room: By monitoring CO2 concentrations, it helps optimize indoor climate control for enhanced comfort and energy efficiency.

2.CO2_(dinning-room): forecast indoor temperature changes. By monitoring CO2 concentrations

3.Relative-humidity_room: The room's relative humidity to forecast indoor temperature variations.

4.Relative_humidity_(dinning-room): to forecast indoor temperature changes.

5.Lighting_room: Lighting conditions to anticipate temperature fluctuations.

6.Meteo_rain: System incorporates meteorological data on rain to forecast indoor temperature changes. By considering rainfall.

7.Meteo_wind: To forecast indoor temperature changes. By analyzing wind conditions.

8. Meteo_Sun_light_in_west_facade: Sunlight exposure on the west facade to forecast indoor temperature changes. By monitoring sunlight.

9.Outdoor_relative_humidity_room: To forecast indoor temperature changes. By analyzing outdoor humidity levels.

10.Lighting_(dinning-room): To forecast indoor temperature variations. By analyzing lighting levels.

11. Meteo_Sun_dusk: Uses data on sunset and dusk times to anticipate indoor temperature changes.

12.Meteo_Sun_light_in_east_facade: Uses data on sunlight exposure on the east facade.

13.Meteo_Sun_light_in_south_facade: South facade optimizes indoor climate control.

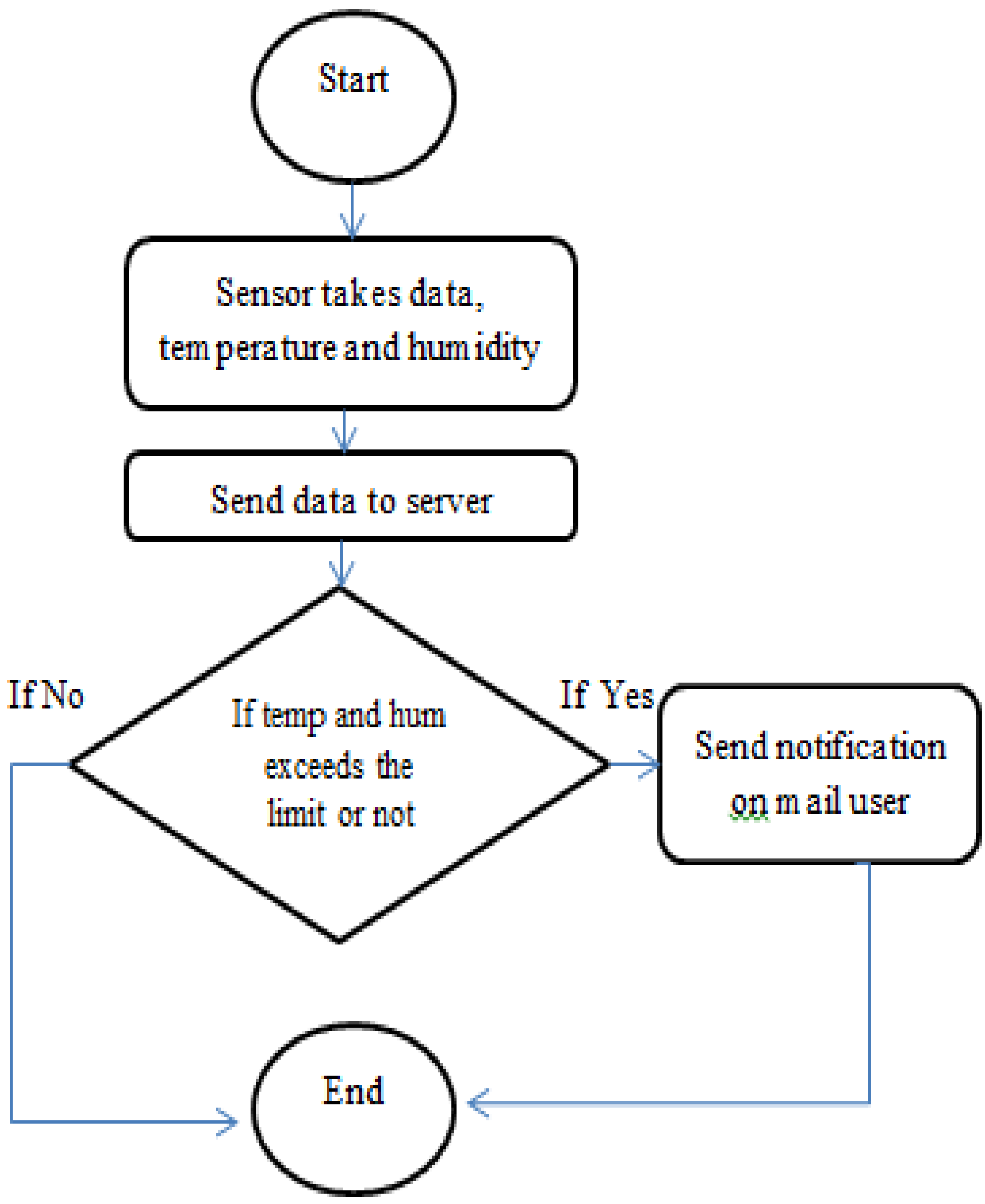
14.Meteo_Sun_irradiance: Solar irradiance data enhances indoor climate.

15.Indoor_room_temperature: Predicts indoor room temperature data.

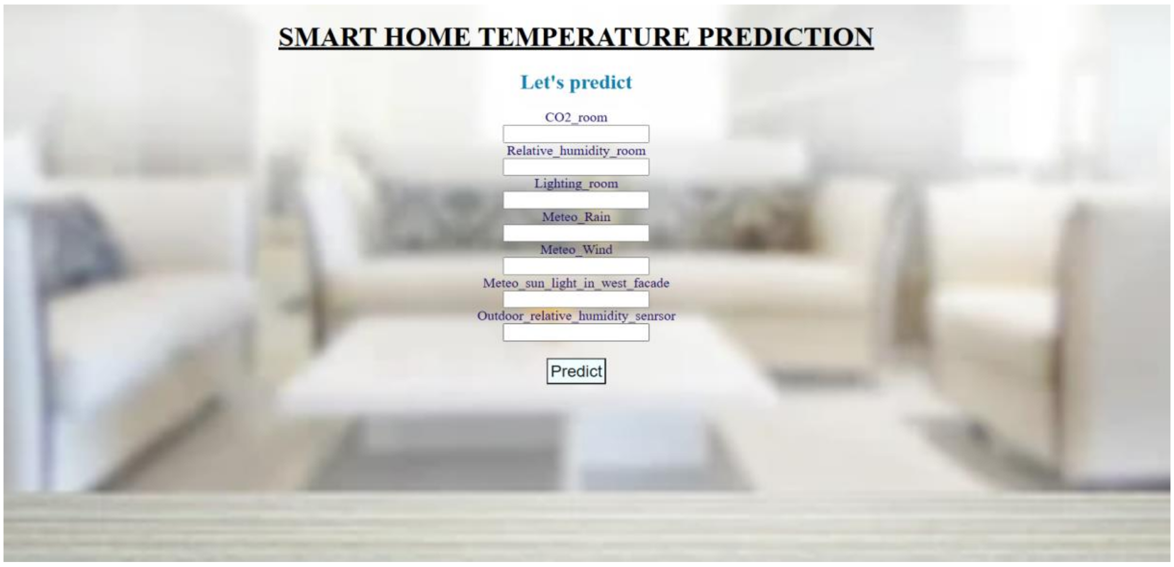
For the dataset we selected, it consists of more than the columns we want to predict it . So, we have chosen the feature drop it contains the columns that we are going to predict the Predtiction value.

- Feature drop means it drops the columns that we don’t want in our dataset.
- Feature_drop = [‘CO2_(dinning-room)’,’Relative_humidity_(dinning-room)’,‘Lighting_(dinning_room)’,‘Meteo_Sun_dusk’,‘Meteo_Sun_light_in_east_facade’,‘Meteo_Sun_light_in_south_facade’,‘Meteo_Sun_irradiance’]

5.FLOWCHART


Start
Sensor takes data, temperature and humidity
Send data to server
If temp and hum exceeds the limit or not
If No
If Yes
Send notification on mail user
End

6.RESULT



SMART HOME TEMPERATURE PREDICTION

Let's predict

CO2_room

Relative_humidity_room

Lighting_room

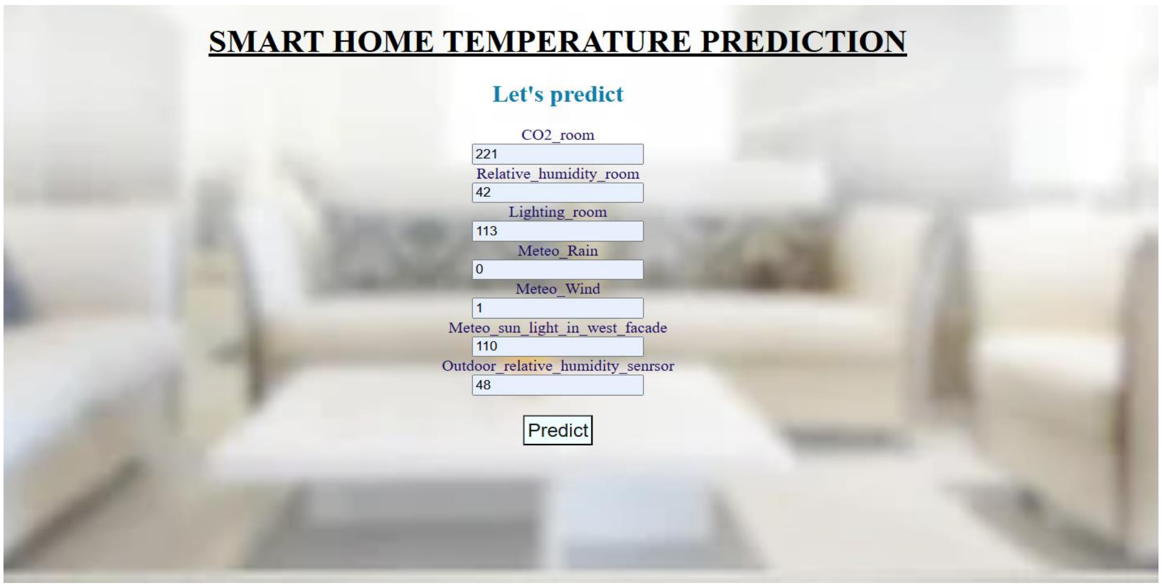
Meteo_Rain

Meteo_Wind

Meteo_sun_light_in_west_facade

Outdoor_relative_humidity_senrsor

Predict



SMART HOME TEMPERATURE PREDICTION

Let's predict

CO2_room: 221

Relative_humidity_room: 42

Lighting_room: 113

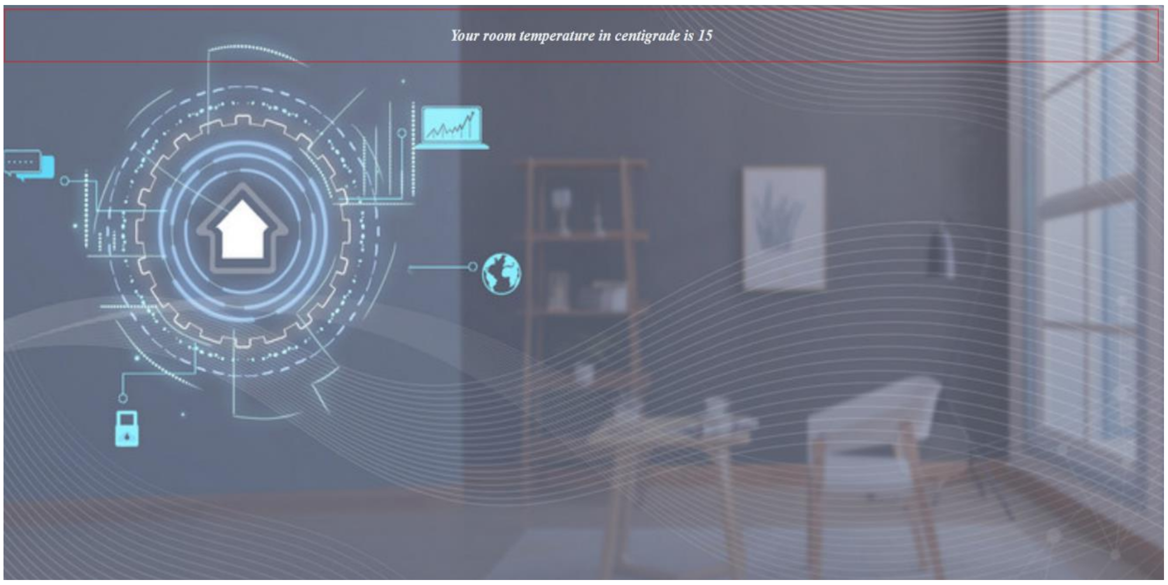
Meteo_Rain: 0

Meteo_Wind: 1

Meteo_sun_light_in_west_facade: 110

Outdoor_relative_humidity_senrsor: 48

Predict



Your room temperature in centigrade is 15

7.ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

1. Energy Efficiency: Optimizes heating and cooling schedules, reducing energy consumption and costs.

2. Comfort: Maintains optimal indoor temperature tailored to occupants' preferences**.**

3. Automation: Reduces the need for manual adjustments by learning and predicting temperature needs.

4. Adaptability: Adjusts to changes in weather, occupancy, and usage patterns.

5. Integration: Can be integrated with other smart home systems (lighting, security) for a comprehensive smart home environment.

DISADVANTAGES:

1. Initial Setup Cost: High cost of installing and setting up smart home systems and machine learning models.

2. Privacy Concerns: Potential risk of data breaches and misuse of personal data.

3. Complexity: Requires technical expertise to set up, manage, and troubleshoot.

4. Reliance on Data: Needs a large amount of data to train the models effectively, which may not always be available.

5. Connectivity Issues: Relies on a stable internet connection; connectivity issues can affect performance.

8.APPLICATIONS

1. Automated Climate Control: Automatically adjusts heating, ventilation, and air conditioning (HVAC) systems to maintain optimal temperatures.

2. Energy Consumption Forecasting: Predicts future energy usage, helping homeowners manage and reduce their energy bills.

3. Smart Thermostats: Enhances the functionality of smart thermostats by learning user preferences and occupancy patterns.

4. Zonal Heating and Cooling: Manages different temperatures in various zones or rooms within a house, improving comfort and efficiency.

5. Integration with Renewable Energy: Aligns temperature control with the availability of renewable energy sources like solar or wind power.

6. Remote Monitoring and Control: Allows homeowners to monitor and control the temperature of their home remotely via mobile apps.

7. Predictive Maintenance of HVAC Systems: Detects anomalies and predicts potential failures in HVAC systems, allowing for proactive maintenance

9.CONCLUSION

Smart home temperature prediction using machine learning offers a promising avenue for enhancing comfort, improving energy efficiency, and reducing costs in residential settings.

By leveraging data and advanced algorithms, these systems can automatically adjust heating and cooling to match user preferences and external conditions, leading to optimized energy use and personalized living environments.

Despite challenges such as initial setup costs, privacy concerns, and the need for continuous data and connectivity, the long-term benefits of reduced energy consumption, increased convenience, and potential for integration with other smart home technologies make it a valuable investment.

As technology continues to advance, the capabilities and efficiencies of these systems are likely to grow, further solidifying their role in sustainable and intelligent home management.

10.FUTURE SCOPE

1.Real-Time Adaptation: Improved real-time data processing capabilities for more responsive and dynamic temperature adjustments.

2. Predictive Maintenance: More sophisticated predictive maintenance for HVAC systems, identifying potential issues before they cause failures.

3. Renewable Energy Integration: Better integration with renewable energy sources, optimizing temperature control based on energy availability and cost.

4. Enhanced Security Features: Advanced security measures to protect user data and ensure privacy in smart home ecosystems.

5. Grid Interaction: Greater interaction with smart grids, enabling homes to become more active participants in energy distribution and demand response programs.

6. Machine Learning Advancements: Utilization of cutting-edge machine learning techniques, such as deep learning and reinforcement learning, for more accurate predictions.

7. Environmental Impact: Contributions to environmental sustainability through more efficient energy use and reduced carbon footprint.

8. Cost Reduction: Lower costs of implementation and maintenance as technology advances and becomes more widespread.

9. User-Friendly Interfaces: Development of more intuitive and user-friendly interfaces for managing smart home temperature settings.

10. Health and Wellbeing: Incorporation of health and wellbeing parameters, adjusting home temperatures to promote better health outcomes.

11.BIBILOGRAPHY

1. Arulmozhi, E., Basak, J.K., Sihalath, T., Park, J., Kim, H.T., Moon, B.E.: Machine learning-based microclimate model for indoor air temperature and relative humidity prediction in a swine building. Animals 11(1), 222 (2021
2. ASHRAE: Guideline 10 provides guidance regarding factors affecting indoor environmental conditions acceptable to the comfort and health of human occupants (2016)
3. Basak, J.K., Okyere, F.G., Arulmozhi, E., Park, J., Khan, F., Kim, H.T.: Artificial neural networks and multiple linear regression as potential methods for modelling body surface temperature of pig. J. Appl. Anim. Res. 48(1), 207–219 (2020)
4. Beghdad, R., Bechar, K., Bouali, M., Haddadi, M.: Neural networks and decision trees for intrusion detections: enhancing detection accuracy. Tech. rep., EasyChair (2020)
5. Chui, K.T., Lytras, M.D., Visvizi, A.: Energy sustainability in smart cities: artificial intelligence, smart monitoring, and optimization of energy consumption. Energies 11(11), 1–20 (2018).
6. Elanchezhian, A., et al.: Evaluating different models used for predicting the indoor microclimatic parameters of a greenhouse. Appl. Ecol. Environ. Res. 18, 2141–2161 (2020)
7. Guedey, M., Uckelmann, D.: Exploring smart home and internet of things technologies for smart public buildings. In: Proceedings of the 10th International Conference on the Internet of Things, pp. 1–8 (2020)
8. He, X., Guan, H., Zhang, X., Simmons, C.T.: A wavelet-based multiple linear regression model for forecasting monthly rainfall. Int. J. Climatol. 34(6), 1898–1912 (2014)
9. Jayalakshmi, T.A.S.: Statistical normalization and back propagation for classification. Int. J. Comput. Theory Eng. (IJCTE) 3, 89–93 (2011)

10. Vassallo, D., Krishnamurthy, R., Sherman, T., Fernando, H.J.S.: Analysis of random forest modeling strategies for multi-step wind speed forecasting. Energies 13(20), 5488 (2020).

11. Lu, T., Viljanen, M.: Prediction of indoor temperature and relative humidity using neural network models: model comparison. Neural Comput. Appl. 18, 345–357 (2009

12. Mohan, P., Patil, K.: Deep learning based weighted SOM to forecast weather and crop prediction for agriculture application. Int. J. Intell. Eng. Syst. 11, 167–176 (2018).

13. Mustafaraj, G., Lowry, G., Chen, J.: Prediction of room temperature and relative humidity by autoregressive linear and nonlinear neural network models for an open office. Energy Build. 43(6), 1452–1460 (2011)

14. Nguyen, T., Fouchereau, R., Frenod, E., Gerard, C., Sincholle, V.: Comparison of forecast models of production of dairy cows combining animal and diet parameters. Comput. Electron. Agric. 170, 105258 (2020).

15. Pérez-Lombard, L., Ortiz, J., Pout, C.: A review on buildings energy consumption information. Energy Build. 40(3), 394–398 (2008)

16. Qi, C., Chang, N.B.: System dynamics modeling for municipal water demand estimation in an urban region under uncertain economic impacts. J. Environ. Manag. 92(6), 1628–1641 (2011)

17. Shi, X., Lu, W., Zhao, Y., Qin, P.: Prediction of indoor temperature and relative humidity based on cloud database by using an improved BP neural network in Chongqing. IEEE Access 6, 30559–30566 (2018)

18. Sola, J., Sevilla, J.: Importance of input data normalization for the application of neural networks to complex industrial problems. IEEE Trans. Nucl. Sci. 44(3), 1464–1468 (1997

19. Tham, K., Ullah, M.: Building energy performance and thermal comfort in Singapore (1993)

20. Traboulsi, S., Knauth, S.: IoT analysis and management system for improving work performance with an IoT open software in smart buildings. J. Ubiquitous Syst. Pervasive Netw. 14(01), 1–6 (2021)

12.APPENDIX

Model building :

1)Dataset
2)Google colab and VS code Application Building
   1. HTML file (Index file, Result file)
   1. Flask file
   2. Models in pickle format

SOURCE CODE:

INDEX.HTML

```
<!DOCTYPE html>
<html lang="en">
 <head>
<meta charset="UTF-8"/>
   <meta name="viewport" content="width=device-width",initial-scale=1.0>
   <link rel="stylesheet" href="style.css" />
 <h1 style="color:black; text-align: center"><u>SMART HOME TEMPERATURE PREDICTION</u></h1>
<style>
     body {
       background: url("static/3.jpg") center;
       height: 100%;
       background-position: center;
       background-size: cover;
       background-repeat: no-repeat;
       position: sticky;
       background-image: url("static/3.jpg");
       background-size: cover;
       background-repeat: no-repeat;
       background-attachment: fixed;
}
```

```
    .container {
        display: flex;
        flex-direction: column;
        align-items: center;
     }
     .input {
        margin: 5px;
    }
    h1 {
      color: rgb(17, 17, 17);
    }
    .btn {
      margin-top: 20px;
      padding: 3px;
      background-color: azure;
      font-size: larger;
      color: rgb(23, 24, 24);
      cursor: pointer;
    }
    form {
      color: rgb(28, 13, 101);
      align-content: center;
      text-align: center
}
</style>
 </style>
  </head>
  <body>
    <h2 style="color: rgb(13, 129, 174); text-align: center">Let's predict</h2>
```

```
<div class="inputs">
    <form action="{{ url_for('predict')}}" method="post">
<label aria-setsize="24px">CO2_room</label><br />
      <input
        type="text"
        name="CO2_room"
       /><br />
       <label>Relative_humidity_room</label><br />
      <input
        type="text"
        name="Relative_humidity_room"
 /><br />
      <label>Lighting_room</label><br />
      <input
        type="text"
        name="Lighting_room"
   /><br />
      <label>Meteo_Rain</label><br />
      <input
        type="text"
        name="Meteo_Rain"
        /><br />
      <label>Meteo_Wind</label><br />
      <input
        type="text"
        name="Meteo_Wind"
       /><br />
      <label>Meteo_sun_light_in_west_facade</label><br />
```

```
        <input type="text’
        name="Meteo_sun_light_in_west_facade"
      /><br />
        <label>Outdoor_relative_humidity_senrsor</label><br />
        <input
          type="text"
          name="Outdoor_relative_humidity_senrsor"
    /><br />
        <a href="result.html"
          ><button class="btn" type="submit">Predict</button></a>
         </form>
     </div>
<br /><br /><section>
      <h3 style="color: blueviolet; text-align: center">  {{ prediction_text }}
      </h3>
 </section>
   </body>
</html>
```

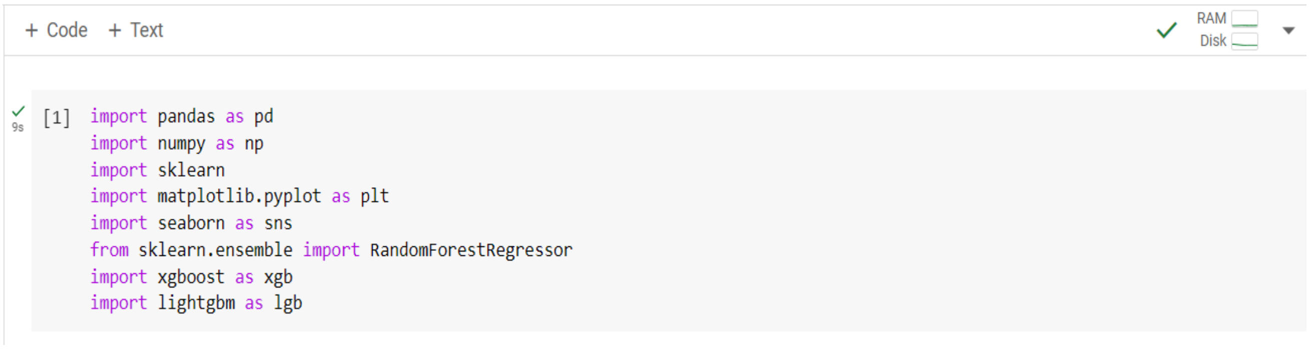
PREDICT.HTML

```
<html>
  <head>
    <title>result</title>
    <style>
      body {
        background-color: darkgrey;
        background-image: url("static/4.jpg");
        background-size: cover;
        background-repeat: no-repeat;
        background-attachment: fixed;
}
```

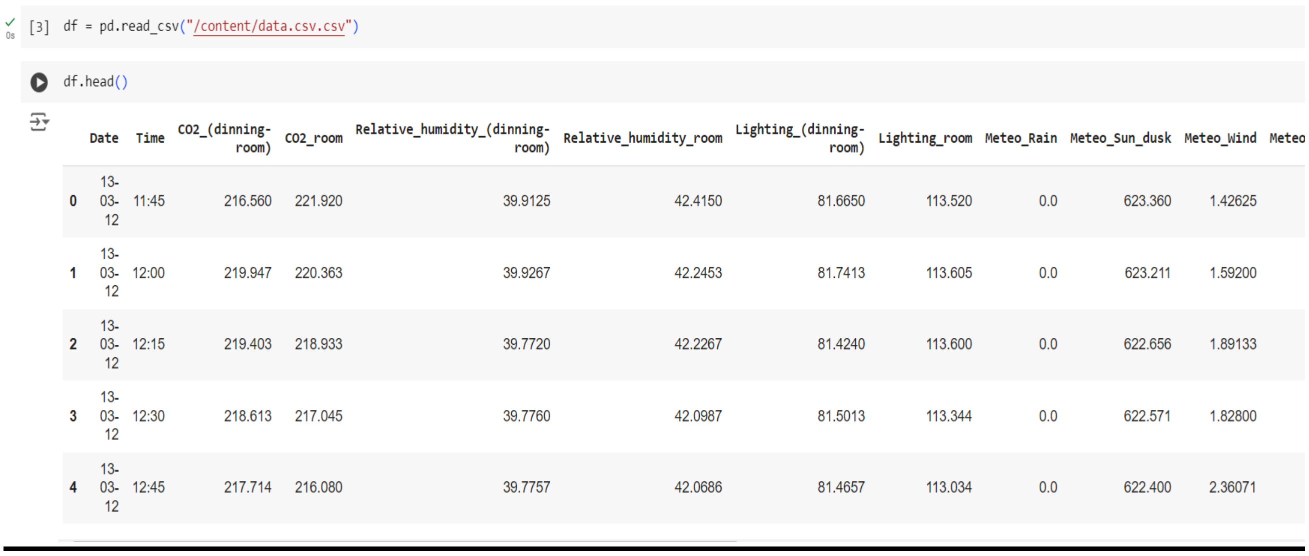
```
      .output {
        padding: 20px;
        border: 1px solid red;
        text-align: center;
        color: rgb(124, 0, 241);
        font-style: italic;
        font-size: larger;
      }
      .result {
        display: block;
        margin-left: auto;
        margin-right: auto;
        width: 50%;
      }
      .centered-text {
         position: absolute;
         top: 50%;
         left: 50%;
         transform: translate(-50%, -50%);
      }
    </style>
  </head>
  <body>
    <h2 class="output" style="color: rgb(234, 238, 239); text-align: center">{{ prediction_text
}}</h2>
  </body>
</html>
```

CODE SNIPPETS

MODEL BUILDING



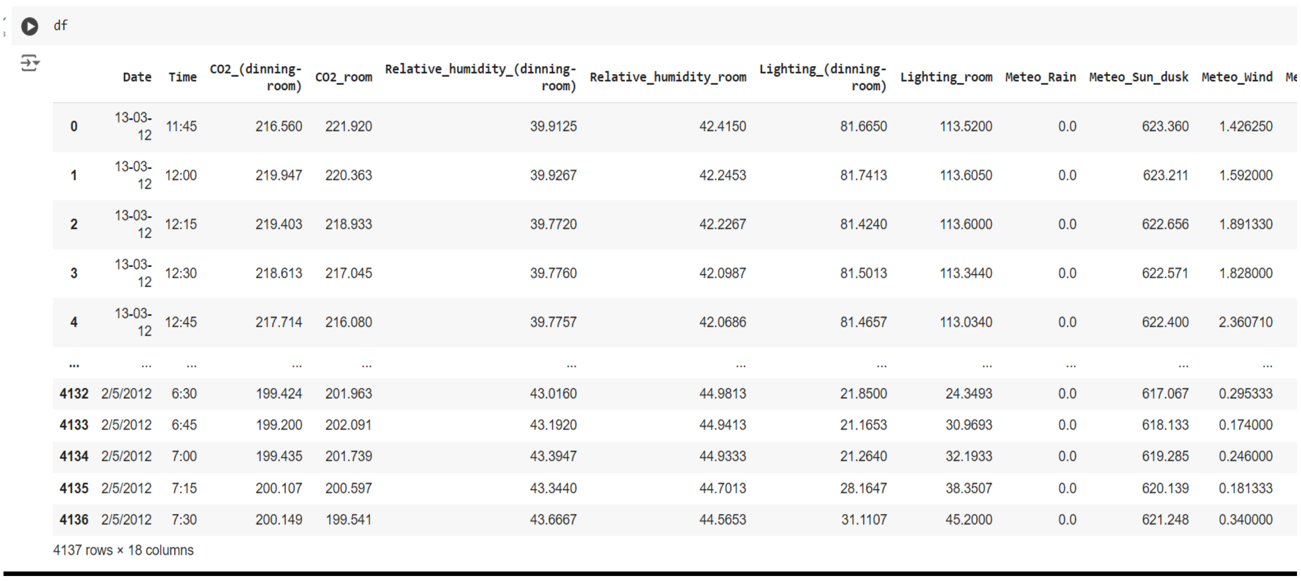
```
[1] import pandas as pd
    import numpy as np
    import sklearn
    import matplotlib.pyplot as plt
    import seaborn as sns
    from sklearn.ensemble import RandomForestRegressor
    import xgboost as xgb
    import lightgbm as lgb
```



```
[3] df = pd.read_csv("/content/data.csv.csv")
```

```
df.head()
```

	Date	Time	CO2_(dinning-room)	CO2_room	Relative_humidity_(dinning-room)	Relative_humidity_room	Lighting_(dinning-room)	Lighting_room	Meteo_Rain	Meteo_Sun_dusk	Meteo_Wind	Meteo
0	13-03-12	11:45	216.560	221.920	39.9125	42.4150	81.6650	113.520	0.0	623.360	1.42625	
1	13-03-12	12:00	219.947	220.363	39.9267	42.2453	81.7413	113.605	0.0	623.211	1.59200	
2	13-03-12	12:15	219.403	218.933	39.7720	42.2267	81.4240	113.600	0.0	622.656	1.89133	
3	13-03-12	12:30	218.613	217.045	39.7760	42.0987	81.5013	113.344	0.0	622.571	1.82800	
4	13-03-12	12:45	217.714	216.080	39.7757	42.0686	81.4657	113.034	0.0	622.400	2.36071	



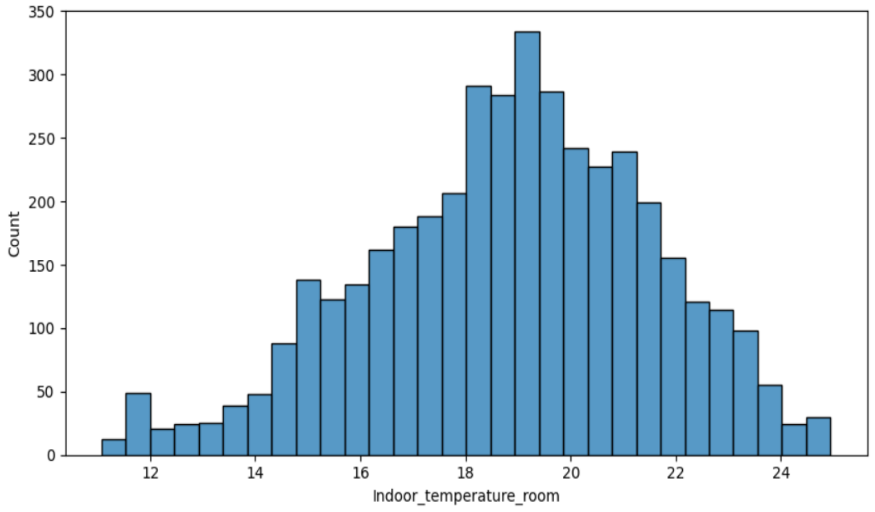
```
df
```

	Date	Time	CO2_(dinning-room)	CO2_room	Relative_humidity_(dinning-room)	Relative_humidity_room	Lighting_(dinning-room)	Lighting_room	Meteo_Rain	Meteo_Sun_dusk	Meteo_Wind	Me
0	13-03-12	11:45	216.560	221.920	39.9125	42.4150	81.6650	113.5200	0.0	623.360	1.426250	
1	13-03-12	12:00	219.947	220.363	39.9267	42.2453	81.7413	113.6050	0.0	623.211	1.592000	
2	13-03-12	12:15	219.403	218.933	39.7720	42.2267	81.4240	113.6000	0.0	622.656	1.891330	
3	13-03-12	12:30	218.613	217.045	39.7760	42.0987	81.5013	113.3440	0.0	622.571	1.828000	
4	13-03-12	12:45	217.714	216.080	39.7757	42.0686	81.4657	113.0340	0.0	622.400	2.360710	
...	...	...	...	...	...	...	...	...	...	...	...	
4132	2/5/2012	6:30	199.424	201.963	43.0160	44.9813	21.8500	24.3493	0.0	617.067	0.295333	
4133	2/5/2012	6:45	199.200	202.091	43.1920	44.9413	21.1653	30.9693	0.0	618.133	0.174000	
4134	2/5/2012	7:00	199.435	201.739	43.3947	44.9333	21.2640	32.1933	0.0	619.285	0.246000	
4135	2/5/2012	7:15	200.107	200.597	43.3440	44.7013	28.1647	38.3507	0.0	620.139	0.181333	
4136	2/5/2012	7:30	200.149	199.541	43.6667	44.5653	31.1107	45.2000	0.0	621.248	0.340000	

4137 rows × 18 columns

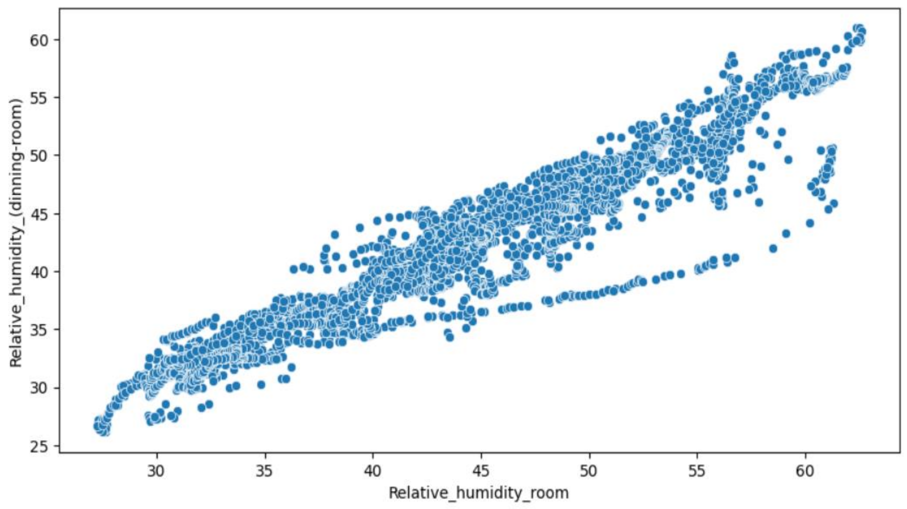
```
[5] plt.figure(figsize =(10,5))
    sns.histplot(data = df, x='Indoor_temperature_room',)
```

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



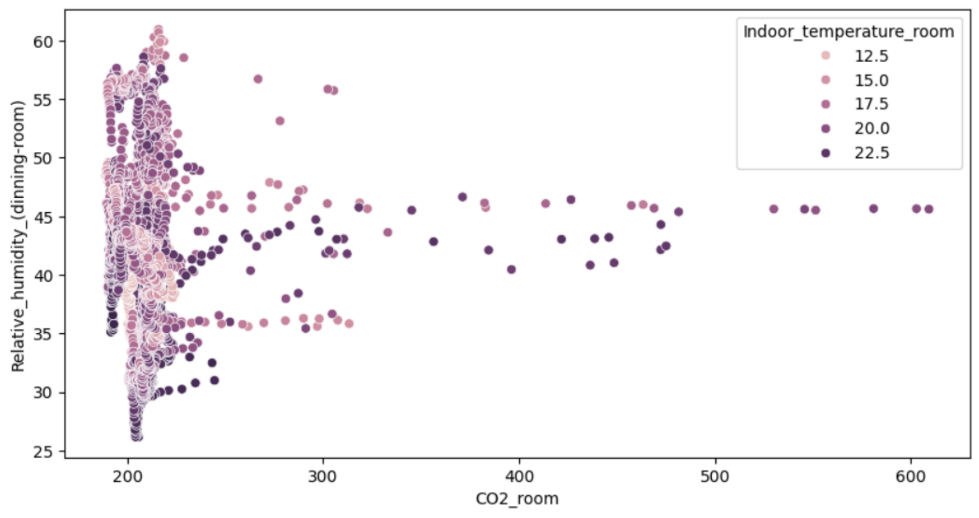
```
plt.figure(figsize =(10,5))
sns.scatterplot(data = df, x= 'Relative_humidity_room', y='Relative_humidity_(dinning-room)')
```

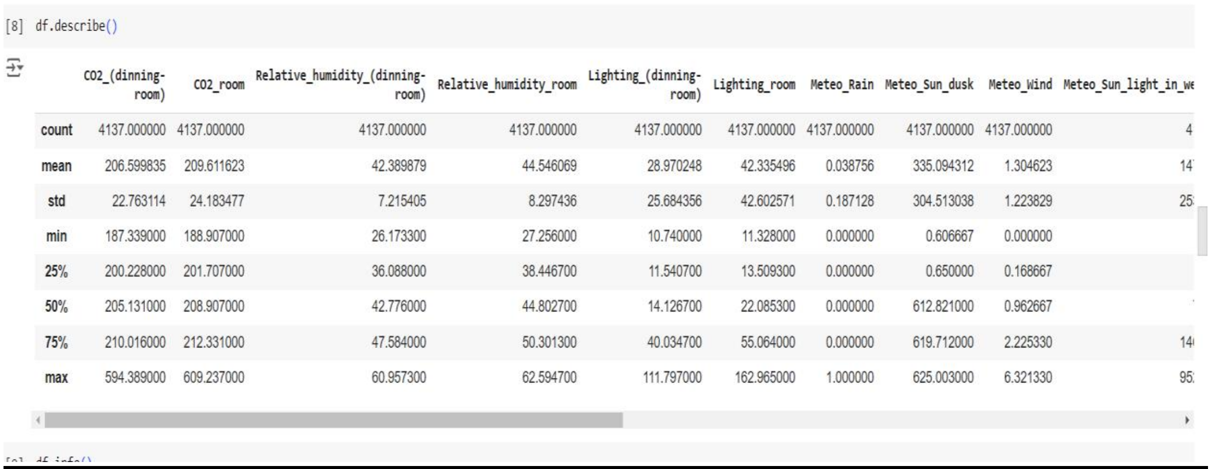
```
<Axes: xlabel='Relative_humidity_room', ylabel='Relative_humidity_(dinning-room)'>
```



```
plt.figure(figsize =(10,5))
sns.scatterplot(data = df, x='CO2_room', y='Relative_humidity_(dinning-room)', hue='Indoor_temperature_room')
```

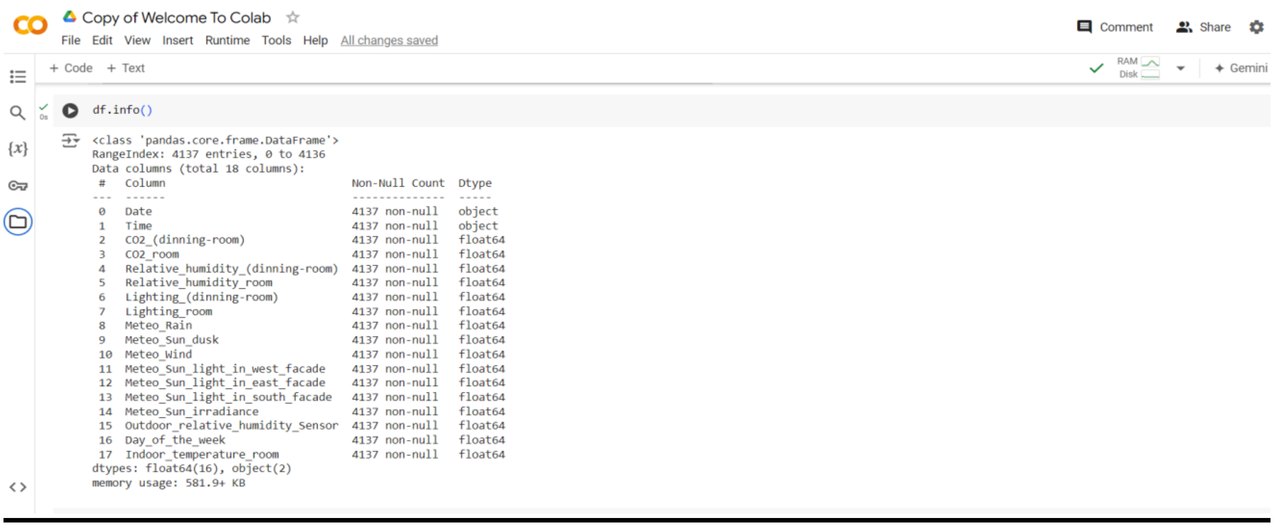
```
<Axes: xlabel='CO2_room', ylabel='Relative_humidity_(dinning-room)'>
```





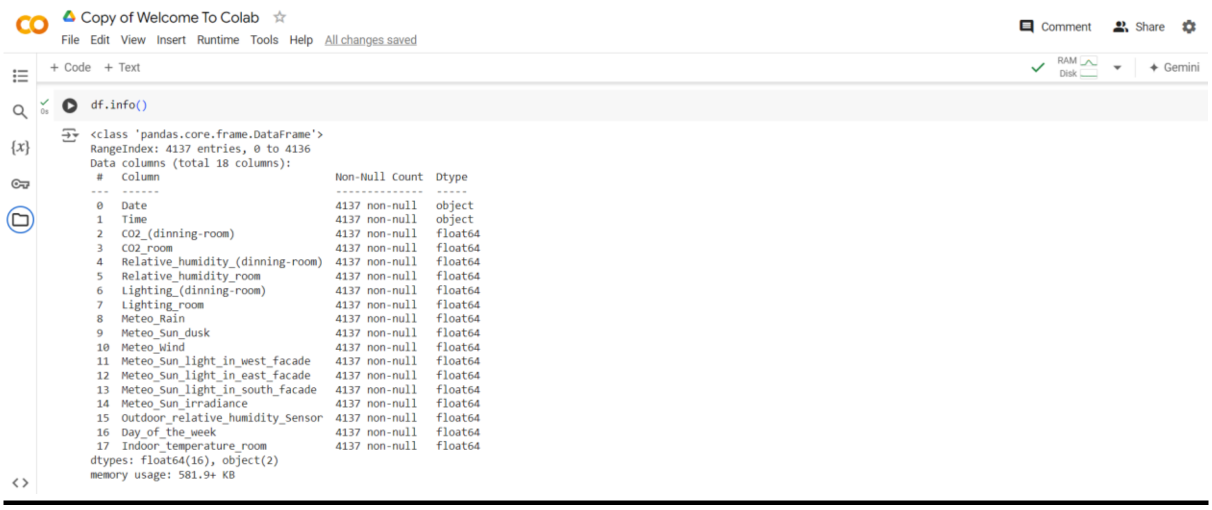
```
[8] df.describe()
```

	CO2_(dinning-room)	CO2_room	Relative_humidity_(dinning-room)	Relative_humidity_room	Lighting_(dinning-room)	Lighting_room	Meteo_Rain	Meteo_Sun_dusk	Meteo_Wind	Meteo_Sun_light_in_we
count	4137.000000	4137.000000	4137.000000	4137.000000	4137.000000	4137.000000	4137.000000	4137.000000	4137.000000	4
mean	206.599835	209.611623	42.389879	44.546069	28.970248	42.335496	0.038756	335.094312	1.304623	14
std	22.763114	24.183477	7.215405	8.297436	25.684356	42.602571	0.187128	304.513038	1.223829	25
min	187.339000	188.907000	26.173300	27.256000	10.740000	11.328000	0.000000	0.606667	0.000000	
25%	200.228000	201.707000	36.088000	38.446700	11.540700	13.509300	0.000000	0.650000	0.168667	
50%	205.131000	208.907000	42.776000	44.802700	14.126700	22.085300	0.000000	612.821000	0.962667	
75%	210.016000	212.331000	47.584000	50.301300	40.034700	55.064000	0.000000	619.712000	2.225330	14
max	594.389000	609.237000	60.957300	62.594700	111.797000	162.965000	1.000000	625.003000	6.321330	95



```
df.info()
```

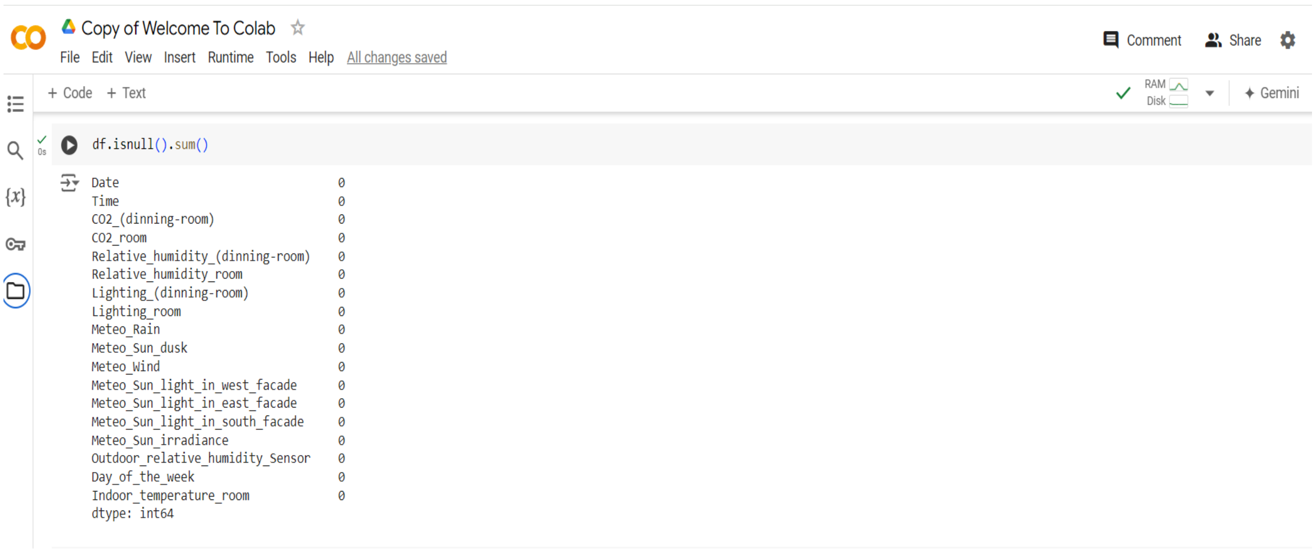
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4137 entries, 0 to 4136
Data columns (total 18 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Date                            4137 non-null   object
 1   Time                            4137 non-null   object
 2   CO2_(dinning-room)              4137 non-null   float64
 3   CO2_room                        4137 non-null   float64
 4   Relative_humidity_(dinning-room) 4137 non-null  float64
 5   Relative_humidity_room          4137 non-null   float64
 6   Lighting_(dinning-room)         4137 non-null   float64
 7   Lighting_room                   4137 non-null   float64
 8   Meteo_Rain                      4137 non-null   float64
 9   Meteo_Sun_dusk                  4137 non-null   float64
 10  Meteo_Wind                      4137 non-null   float64
 11  Meteo_Sun_light_in_west_facade  4137 non-null   float64
 12  Meteo_Sun_light_in_east_facade  4137 non-null   float64
 13  Meteo_Sun_light_in_south_facade 4137 non-null   float64
 14  Meteo_Sun_irradiance            4137 non-null   float64
 15  Outdoor_relative_humidity_Sensor 4137 non-null  float64
 16  Day_of_the_week                 4137 non-null   float64
 17  Indoor_temperature_room         4137 non-null   float64
dtypes: float64(16), object(2)
memory usage: 581.9+ KB
```



Copy of Welcome To Colab

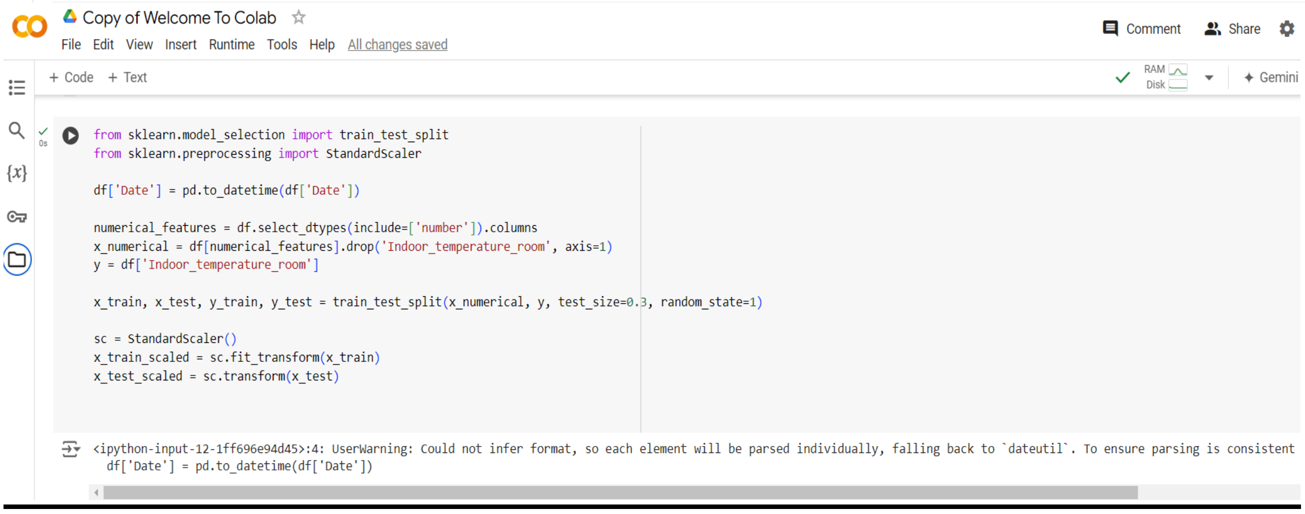
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4137 entries, 0 to 4136
Data columns (total 18 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Date                            4137 non-null   object
 1   Time                            4137 non-null   object
 2   CO2_(dinning-room)              4137 non-null   float64
 3   CO2_room                        4137 non-null   float64
 4   Relative_humidity_(dinning-room) 4137 non-null  float64
 5   Relative_humidity_room          4137 non-null   float64
 6   Lighting_(dinning-room)         4137 non-null   float64
 7   Lighting_room                   4137 non-null   float64
 8   Meteo_Rain                      4137 non-null   float64
 9   Meteo_Sun_dusk                  4137 non-null   float64
 10  Meteo_Wind                      4137 non-null   float64
 11  Meteo_Sun_light_in_west_facade  4137 non-null   float64
 12  Meteo_Sun_light_in_east_facade  4137 non-null   float64
 13  Meteo_Sun_light_in_south_facade 4137 non-null   float64
 14  Meteo_Sun_irradiance            4137 non-null   float64
 15  Outdoor_relative_humidity_Sensor 4137 non-null  float64
 16  Day_of_the_week                 4137 non-null   float64
 17  Indoor_temperature_room         4137 non-null   float64
dtypes: float64(16), object(2)
memory usage: 581.9+ KB
```



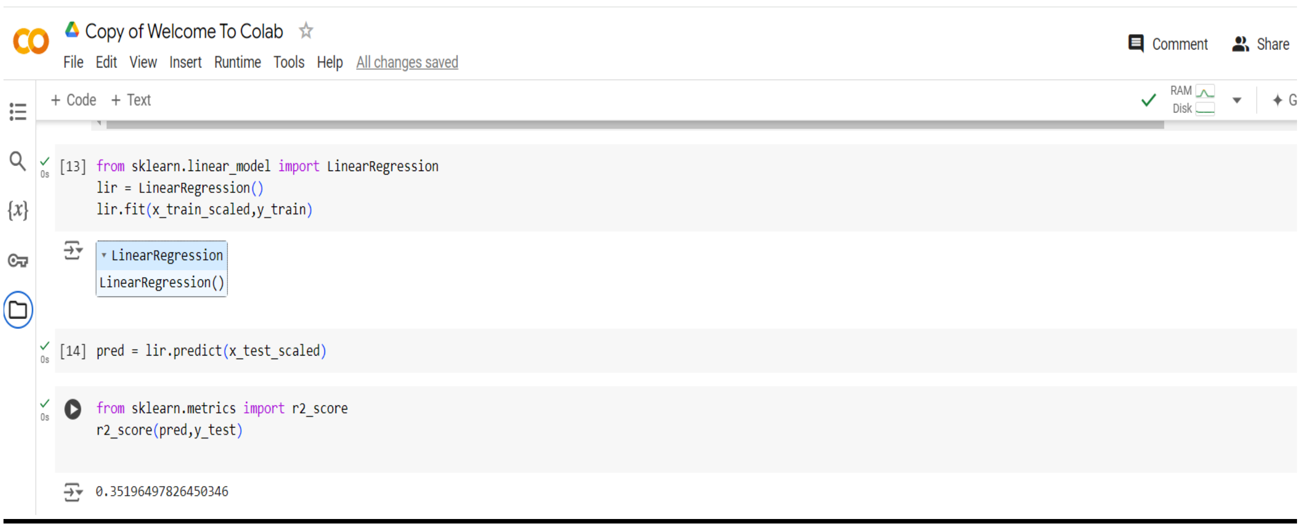
```
df.isnull().sum()
```

```
Date                                  0
Time                                  0
CO2_(dinning-room)                    0
CO2_room                              0
Relative_humidity_(dinning-room)      0
Relative_humidity_room                0
Lighting_(dinning-room)               0
Lighting_room                         0
Meteo_Rain                            0
Meteo_Sun_dusk                        0
Meteo_Wind                            0
Meteo_Sun_light_in_west_facade        0
Meteo_Sun_light_in_east_facade        0
Meteo_Sun_light_in_south_facade       0
Meteo_Sun_irradiance                  0
Outdoor_relative_humidity_Sensor      0
Day_of_the_week                       0
Indoor_temperature_room               0
dtype: int64
```

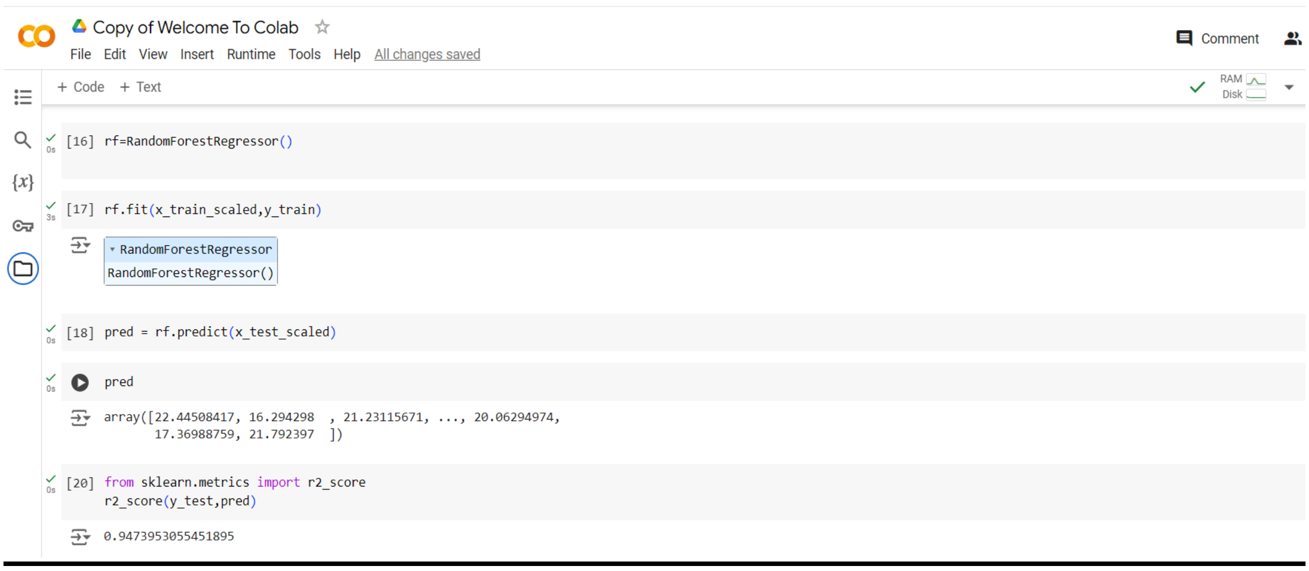


```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

df['Date'] = pd.to_datetime(df['Date'])

numerical_features = df.select_dtypes(include=['number']).columns
x_numerical = df[numerical_features].drop('Indoor_temperature_room', axis=1)
y = df['Indoor_temperature_room']

x_train, x_test, y_train, y_test = train_test_split(x_numerical, y, test_size=0.3, random_state=1)

sc = StandardScaler()
x_train_scaled = sc.fit_transform(x_train)
x_test_scaled = sc.transform(x_test)
```

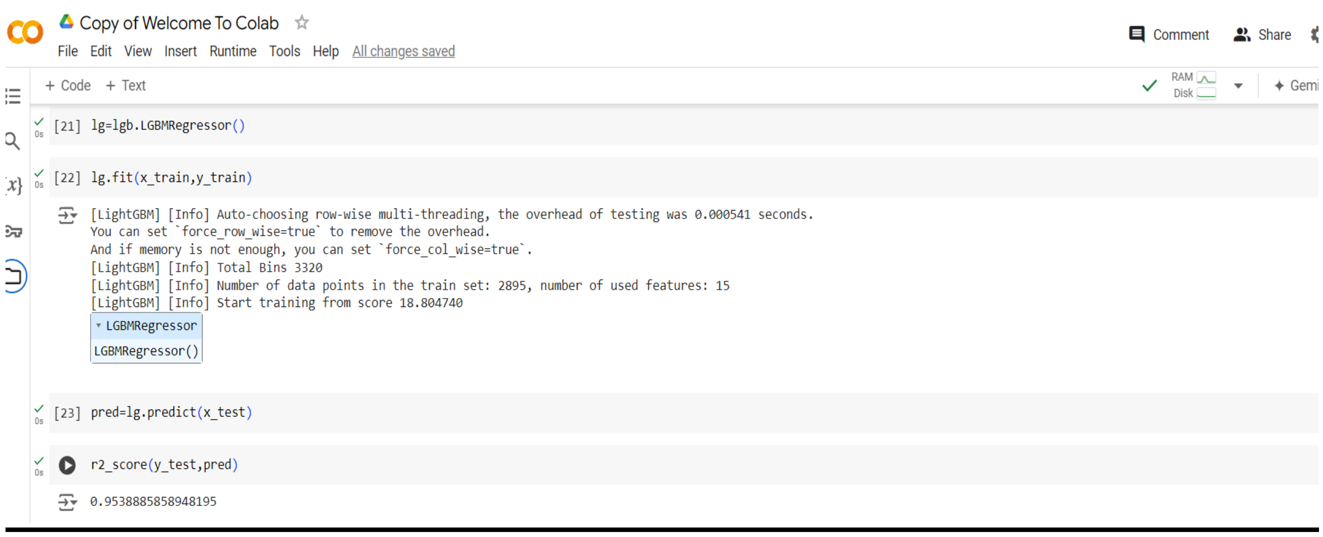
```
<ipython-input-12-1ff696e94d45>:4: UserWarning: Could not infer format, so each element will be parsed individually, falling back to `dateutil`. To ensure parsing is consistent
  df['Date'] = pd.to_datetime(df['Date'])
```



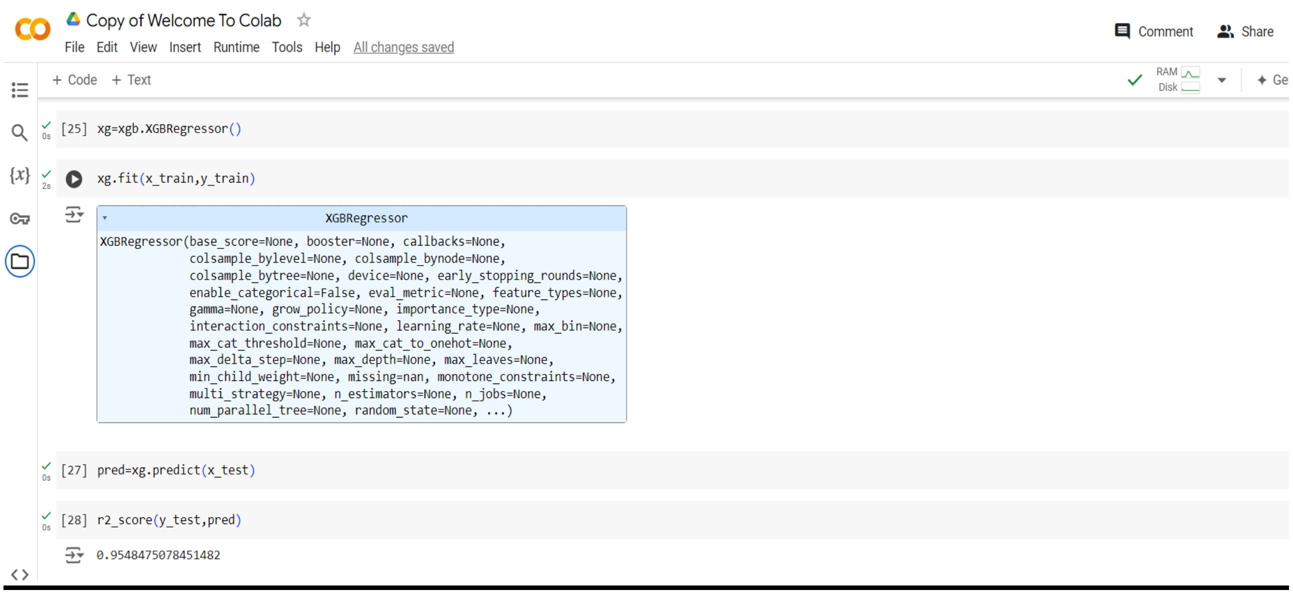
```
[13] from sklearn.linear_model import LinearRegression
     lir = LinearRegression()
     lir.fit(x_train_scaled,y_train)

  ▾ LinearRegression
    LinearRegression()

[14] pred = lir.predict(x_test_scaled)

     from sklearn.metrics import r2_score
     r2_score(pred,y_test)

  0.35196497826450346
```



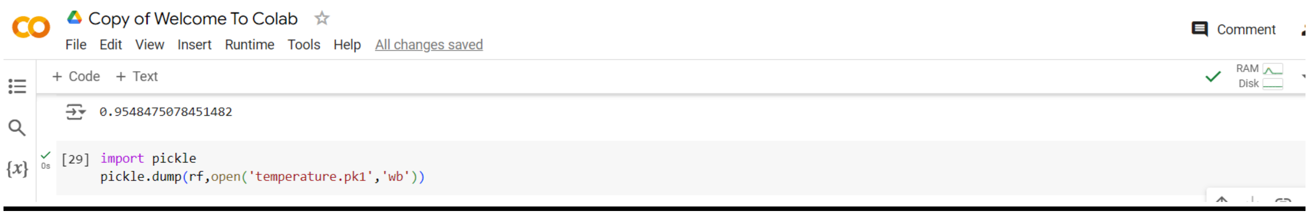
```
[16] rf=RandomForestRegressor()

[17] rf.fit(x_train_scaled,y_train)

  ▾ RandomForestRegressor
    RandomForestRegressor()

[18] pred = rf.predict(x_test_scaled)

     pred

  array([22.44508417, 16.294298  , 21.23115671, ..., 20.06294974,
         17.36988759, 21.792397  ])

[20] from sklearn.metrics import r2_score
     r2_score(y_test,pred)

  0.9473953055451895
```



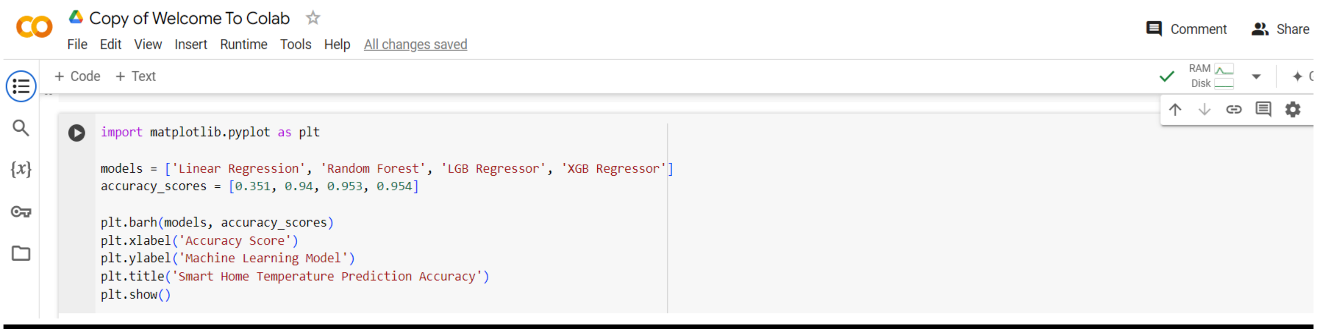
```
[21] lg=lgb.LGBMRegressor()

[22] lg.fit(x_train,y_train)

  [LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.000541 seconds.
  You can set `force_row_wise=true` to remove the overhead.
  And if memory is not enough, you can set `force_col_wise=true`.
  [LightGBM] [Info] Total Bins 3320
  [LightGBM] [Info] Number of data points in the train set: 2895, number of used features: 15
  [LightGBM] [Info] Start training from score 18.804740
  ▾ LGBMRegressor
    LGBMRegressor()

[23] pred=lg.predict(x_test)

     r2_score(y_test,pred)

  0.9538885858948195
```



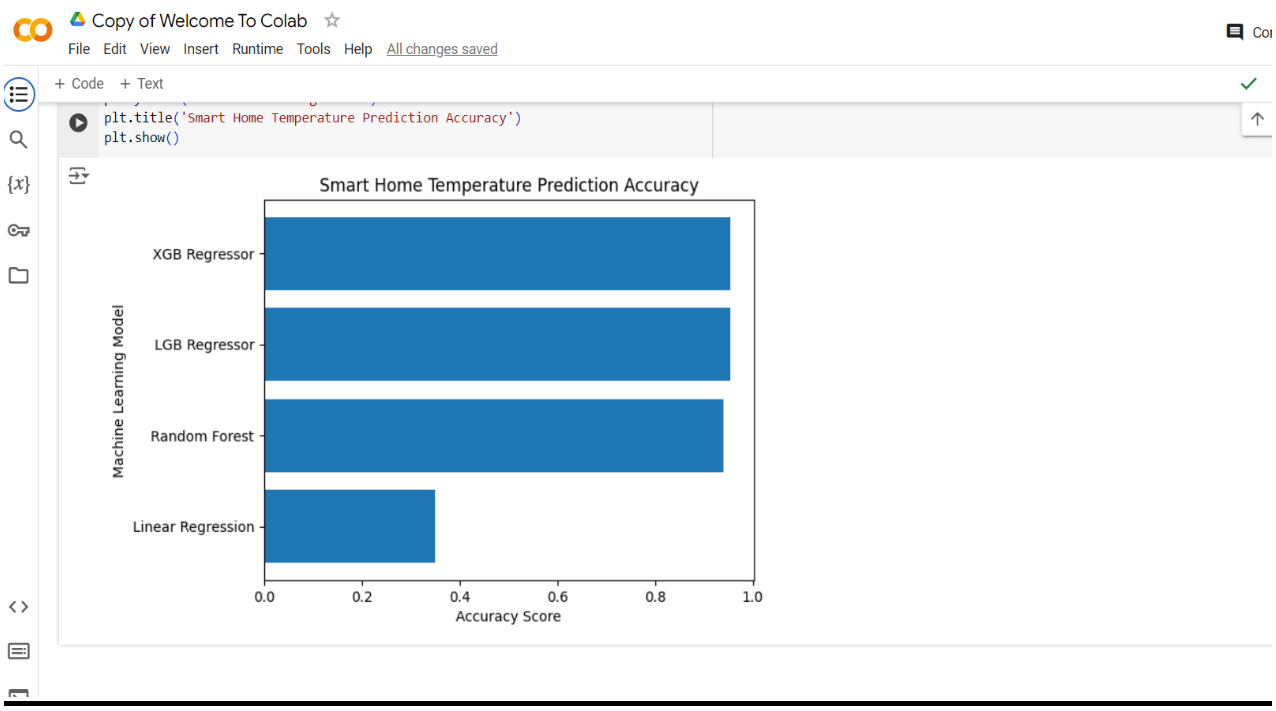
```
[25] xg=xgb.XGBRegressor()

     xg.fit(x_train,y_train)

     XGBRegressor
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=None,
             num_parallel_tree=None, random_state=None, ...)

[27] pred=xg.predict(x_test)

[28] r2_score(y_test,pred)

0.9548475078451482
```



```
0.9548475078451482

[29] import pickle
     pickle.dump(rf,open('temperature.pk1','wb'))
```



```
import matplotlib.pyplot as plt

models = ['Linear Regression', 'Random Forest', 'LGB Regressor', 'XGB Regressor']
accuracy_scores = [0.351, 0.94, 0.953, 0.954]

plt.barh(models, accuracy_scores)
plt.xlabel('Accuracy Score')
plt.ylabel('Machine Learning Model')
plt.title('Smart Home Temperature Prediction Accuracy')
plt.show()
```



```
plt.title('Smart Home Temperature Prediction Accuracy')
plt.show()
```