Analyzing Data from HVAC Systems in Oslo School Buildings with Artificial Intelligence

IKT590 Master Thesis Assignment
May 2024

Analyzing Data from HVAC Systems in Oslo School Buildings with Artificial Intelligence

Under the guidance of

Professor Daniel Romero Gonzalez

And

External project coordinator Sahdat Saleem Malik

Study and project presentation by Niladri Malakar

Candidate no:427

Innhold

Abstract

Artificial intelligence (AI) has significantly altered our lives and will do so in the future. Among the several applications and industries that use it is building automation. Productivity has increased because of artificial intelligence's broad use and close ties to society and industry.

A major development in building automation is the coming together of Artificial Intelligence (AI) and HVAC (heating, ventilation, and air conditioning) systems. This abstract examines how AI algorithms and HVAC technology work together, highlighting how they have the power to completely transform system performance, occupant comfort, and energy efficiency. Advanced algorithms are used by AI-driven HVAC systems to evaluate enormous volumes of sensor data, weather forecasts, occupancy patterns, and past usage. These systems enhance heating and cooling operations to dynamically respond to changing conditions while decreasing energy use through real-time monitoring and predictive analytics.

Energy Efficiency, using predictive modeling, AI algorithms allow proactive HVAC setting adjustments that maximize energy efficiency without sacrificing comfort.

Predictive maintenance reduces downtime and lengthens system lifespan by using artificial intelligence (AI) to analyze equipment performance data, forecast probable breakdowns, and schedule maintenance operations in advance.

Personalized Comfort, AI-enabled HVAC systems can recognize user preferences and modify airflow and temperature to optimize energy savings while improving personal comfort.

Adaptive Control: By continuously learning from feedback on system performance, machine learning algorithms adjust operating parameters to achieve optimal performance under a range of circumstances.

There is great potential for increasing productivity, cutting expenses, and raising occupant happiness through the integration of AI into building HVAC systems.

These abstract lays the groundwork for future research on the usefulness and benefits of AI-driven HVAC technologies in the real world.

Preface

This thesis writing is a part of master course with IKT at University of Agder. The idea behind writing thesis came from current challenges of Oslobygg with HVAC systems. This thesis mostly focused on ***application-oriented thoughts and understanding possibilities with real HVAC sensor data*** from last 3 years. This is already an active industry project and Oslobygg is one of the largest building manufacturer organizations in Norway.

There are different challenges that can be identified, and solution can be planned once we start data analysis and figure out checkpoints to start with. Data collection and then to know the data is the key to start the AI analysis and predictions.

Eventually it will be converted into one project using which many different challenges can be encountered with AI. Considering the next stage I have proposed and designed Datawarehouse and data transformation aspects and that is one step before we have a row dataset ready.

I tried to check the data validity with different ways just to cross check if the expected situations have fulfilled over last 3 years' time, also used different algorithms to understand the actual scenario compared to environment changes and system settings changes. Though it's not enough to just say it is expected result. I must talk to subject experts with HVAC and test results will be pointer to explore new discussion in very recent future.

I had many one-to-one meetings with subject experts and understand the different configurations Oslogygg uses today. I build three different datasets but used only one due to shortage of time, but we have many more scope to develop in the future. Considering 2000 buildings we have huge data from many rooms to be explored.

I can say this is just the start and this project will be extended with larger scale within Oslo Kommune and AI knowledge will be useful even to use standard AI service providers like Microsoft.

On a parallel path being am IKT employee of oslobygg, I will try to use Microsoft AI solutions (standard and custom if any) to learn more on this area.

Acknowledgements

I would want to express my gratitude and regards to a few individuals for their assistance in producing my thesis.

I want to start by expressing my gratitude to my supervisor, professor Daniel Romero Gonzalez, for his excellent guidance and thought-provoking discussions during the semester.

I would like to thank my co supervisor Sahdat Saleem Malik (from Oslobygg) for his support with time management and data collection and idea development.

I would like to thank Dag Ronald Nybakk by reporting manager in Oslobygg who helped me approving the required time and necessary budget to talk to several people to reach to the data source after talking directly to vendors.

I would like to thank Ståle Grihamar to reach correct responsible people who knows more about data and can help me with raw data. I also thank Oddgeir Haug, Håvard Ellann
 and Eirik Fotland for their time and explaining the actual situations and HVAC configurations in Oslobygg.

Background

The Oslo municipality owns Oslobygg KF, a municipal corporation established to develop, own, and manage the municipality's purpose-built buildings for effective service delivery. In compliance with business principles, the organization must oversee the true values that the properties represent and guarantee the regular and effective upkeep and development.

The real estate firm for the Oslo municipality, Oslobygg is among the biggest developers and property managers in the nation. Its social objective is to maintain, develop, and construct the municipality's purpose-built properties while keeping the needs of the locals in mind. Their holdings include national institutions in the capital as well as nurseries, schools, care facilities, nursing homes, cultural structures, sports facilities, and fire stations. **[1]**

In Oslobygg's portfolio, they have numerous fascinating projects and over 2.7 million square meters of property. Over the next few years, they plan to invest around NOK 35 billion. They dream to have to be the engine of a serious, inclusive, and sustainable industry. Optimizing building automation performance, improving occupant comfort, and lowering energy consumption and operating costs are the main objectives of building automation.

Building automation began to take shape in the middle of the 20th century when lighting and temperature management were handled by basic control systems. Building automation has evolved into complex systems with real-time monitoring, data analysis, and adaptive control over time because to developments in computing, sensor, and communication protocols. Oslobygg uses and implementing HVAC systems for different buildings and rooms.

[2]

The HVAC system is composed of three distinct parts:

H stands for heating, which oversees keeping the space warm and cozy in the winter.

V stands for ventilation, which is the process of removing old air from the space and introducing fresh air. By lowering pollutants like CO2 and regulating humidity, it preserves healthy indoor air quality.

AC stands for air conditioning, which cools a space in warm weather.

Numerous exergy analytical techniques have been developed recently for a range of applications, such as operational diagnosis, local optimization, and physical criterion assessment. Numerous exergy analytical techniques have been developed recently for a range of applications, such as operational diagnosis, local optimization, and physical criterion assessment.

Objective

Most studies conducted in the last several decades on indoor air quality (IAQ) have focused on the importance of temperature and CO2 measurements and how to use these data to regulate ventilation.

Regarding the advantages of sustainability, everyone concerns. Cutting back on energy use also has positive economic effects. Lowering energy use also has positive economic effects.

There's the issue of creativity. The HVAC solutions that are currently being offered offer a significant amount of energy reduction, but soon, new methods and AI controlled analytics will help in this area. Beyond lowering CO2 emissions, sustainable thinking considers the whole product cycle. [3]

This thesis is the observation, understanding and planning to roll out massive project within Oslobygg, in coming future. Where data collection must be automated from different sources and data conversion, modification and cleaning should be automatic with different relevant tools or modern standard technology. Building stable Datawarehouse to generate data set for AI on runtime is one of the key aspects.

Studying historical data points and usage of the system components are focus area. It has been noticed that some heating radiator and cooling systems are running parallel and that is kind of mal function or wrong configuration of the HVAC system settings. Random and real time data analysis and deep level monitoring will improve the situation and will add value to the society.

Oslobygg manages all schools, kindergarten, Hospitals and Swimming pools, sports arenas so the activity is huge. These building HVAC systems are sometimes based on very old technology, and some are recently modernized.

This study will give an overall performance analysis and help to track the energy reduction strategies.

Introduction

Evaluating the design, operation, efficiency, and performance of HVAC (heating, ventilation, and air conditioning) systems entails several different considerations. HVAC systems have various modes of operation for ventilation, heating, and cooling.

Analyze the system's ventilation. This entails evaluating the ductwork's configuration and construction, the functionality of fans or blowers, and the way that conditioned air is distributed throughout the structure. Ineffective operation and comfort problems might result from poor ventilation.

Examine the control mechanisms that govern how HVAC equipment operates. This comprises programmable logic controllers (PLCs), sensors, zoning systems, and thermostats. It is crucial to comprehend how these systems cooperate to preserve efficiency and comfort.

Think about how HVAC systems affect the quality of the air indoors. Discover the necessity of adequate ventilation in maintaining a healthy interior environment and eliminating contaminants, as well as the needs for air filtration, humidity management, and ventilation.

Examine the effects of HVAC systems on the environment, considering their capacity to cut greenhouse gas emissions as well as their energy and refrigerant usage. Seek out chances to use alternative technologies and system modifications to increase efficiency and lessen environmental damage. The objective is to have a thorough understanding of HVAC systems and learn how to evaluate them for comfort, efficiency, and performance.

To reduce the possibility of viruses spreading through the air, indoor CO2 levels should be measured at a particular threshold. The ideal range for outdoor CO2 concentrations is between 400 and 800 (PPM) parts per million. It is advised to ventilate the area, exit the room, and replace the air if the threshold is surpassed. [4]

However, current ventilation rules, such those from the American Society of Heating, Refrigerating, and Air Conditioning Engineers (ASHRAE), advise not exceeding the concentration of outdoor air in the area by more than 650 parts per million (ppm) indoors.

Literature review

Working with HVAC system data is important. By discovering residents' preferences and modifying HVAC settings accordingly, AI can personalize interior comfort. Artificial intelligence (AI) systems can optimize temperature and humidity settings based on user preferences and maximize energy efficiency by evaluating historical data and real-time feedback.

AI systems can examine data from HVAC systems to find patterns and anticipate equipment breakdowns before they happen. Based on sensor data, artificial intelligence techniques like machine learning may automatically identify flaws and fix issues. This can assist facility managers in quickly identifying problems and implementing fixes to guarantee the best possible system performance.

Holistic building management is made possible by the integration of HVAC system data with other building systems, including lighting, occupancy sensors, and security. Artificial intelligence (AI)-powered platforms can coordinate the way various systems interact to maximize building efficiency, improve occupant comfort, and simplify operations.

By evaluating data like as temperature, humidity, occupancy patterns, and external weather conditions, AI can enhance these systems' performance and make dynamic setting adjustments. Significant energy savings and a smaller environmental effect may result from this.

There are some recent studies and experiment papers published in 2024 on this topic.

Published on recent related papers in this context.

- Prediction and optimization of building ventilation and heat loads. ANFIS model for precise prediction of HVAC performance. [5]
- AI for diagnosing and detecting HVAC problems. [6]
- Energy Efficiency and Patient Comfort in Hospitals with AI and IOT Integration [7]
- Examining Machine Learning Algorithms for Sustainability and Efficiency in HVAC [8]
- Deep reinforcement learning control using predictive information. This case study is centered on the HVAC system of an office building. [9]
- Evaluating the Effects of Climate Change on Project Management and HVAC System [10]
- Enhanced chiller system energy management with AI-based regression [11]
- Modeling HVAC control methods for buildings with a deep reinforcement learning methodology. [12]

Aim of the Study

Oslo is capital of in Norway located in in North of Europe and with an average daily high temperature above 16°C, the warm season spans 3.4 months, from middle of May to early September. July is the warmest month of the year in Oslo, with typical highs of 22°C and lows of 12°C.

With an average daily high temperature below 2°C, the cold season spans 3.8 months, from third week of November to middle March. January is Oslo's coldest month of the year, with an average low temperature of -7°C and high temperature of -1.6°C. [13]

Oslobygg uses HAVC systems as these systems are mandatory to use in Oslo school buildings. It is very useful to study how efficiently all these systems are performing. How performance can be improved.

Energy Efficiency: Automation can minimize energy usage and utility costs by optimizing HVAC, lighting, and other building systems.

Enhanced Comfort: The best possible levels of comfort for building occupants are guaranteed by automated environmental condition control.

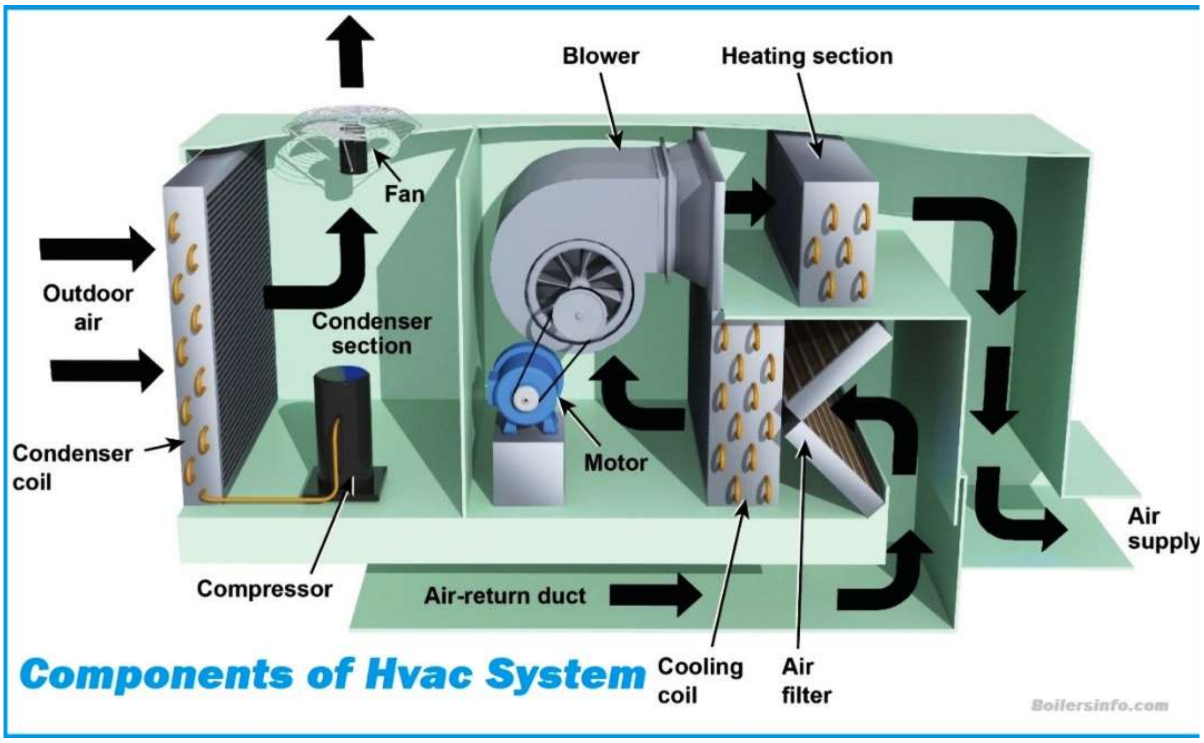
Remote Monitoring and operate: Proactive maintenance and troubleshooting are made possible by the ability of building operators to remotely monitor and operate equipment.

Enhanced Security and Safety: Automation systems can be integrated with fire alarm and security systems to improve emergency response times and building safety in future.

Sustainability: Building automation helps achieve sustainability objectives and green building certifications by lowering energy consumption and greenhouse gas emissions.

All things considered, building automation is essential to contemporary building management since it provides a complete answer for increasing occupant contentment, cutting expenses, and optimizing performance. One way this study checks the existing effectiveness of the system and on the other hand this will become the base for further discussing points to focus on new research or automation area in coming future.

HVAC system components



Figur 1: Components HVAC System [0]

Considering the initial situation in this project, historical data has been collected from vendors. This data converted into dataset and some of the key components are there. It is important to be familiar of these components as they are obvious tools and operating together to maintain the indoor usable.

Radiator valve position value

A numerical or categorical value that indicates the position or setting of a radiator valve is sometimes referred to as the "radiator valve position value". A room's or area's temperature can be controlled by using radiator valves to manage the hot water flow via a radiator.

Depending on the kind of radiator valve, the position value may change. A numerical scale or a range of predetermined positions, such as "0" for entirely closed and "5" for fully open, may be used for manual valves. A temperature setpoint or a relative position between fully closed and fully open may be represented by the position value for thermostatic radiator valves (TRVs), which automatically adjust in response to ambient temperature.

Monitoring the position of the radiator valve in the context of HVAC (heating, ventilation, and air conditioning) systems or building automation can be helpful for maximizing energy utilization, preserving comfort levels, and identifying possible problems with heating systems. It makes it possible for automated control systems or building managers to effectively modify the heating levels according to occupancy, outside temperature, and other variables.

Supply Air VAV Air Volume Flow

The term "Supply Air VAV Air Volume Flow" describes the air volume flow rate in the supply duct of a Variable Air Volume (VAV) system.

Commercial buildings often use Variable Air Volume (VAV) systems as part of their HVAC (Heating, Ventilation, and Air Conditioning). These systems change the amount of air supplied to different rooms or zones to control the airflow. The volume of air provided to a specific location via the supply ductwork is indicated by the supply air volume flow rate.

According to the heating or cooling requirements of the zone they service, VAV systems are made to change the airflow. Usually, to create this variation, damper positions or fan speeds are modulated. Maintaining the building's interior air quality, energy efficiency, and comfort levels requires constant monitoring and management of the supply air volume flow rate.

The supply air volume flow rate is frequently measured and dynamically regulated in building automation systems using sensors and control algorithms in response to variables including occupancy, temperature, and ventilation requirements. The effectiveness and efficiency of the HVAC system are enhanced by proper monitoring and control of this parameter.

Extract Air VAV Air Volume Flow

The term "Extract Air VAV Air Volume Flow" describes the air volume flow rate in the extract or return air duct of a Variable Air Volume (VAV) system.

By varying the volume of air supplied and removed, VAV systems are frequently used in HVAC systems in commercial buildings to control airflow to different zones or rooms. The amount of air extracted from a particular region via the return or extract ductwork is indicated by the extract air volume flow rate.

To preserve indoor air quality and thermal comfort, air in a VAV system is normally removed from each zone and recirculated throughout the structure. It is possible to modify the extract air volume flow rate in accordance with standards of ventilation, indoor air quality requirements, and occupancy levels.

It is essential to monitor and regulate the extract air volume flow rate to keep the building's energy efficiency, indoor air quality, and appropriate ventilation. Building automation systems frequently measure and adjust this parameter dynamically using sensors and control algorithms to make sure the HVAC system functions well and satisfies the needs of the occupants in terms of comfort and health.

Plant Operating Mode

The term " Plant Operating Mode" describes how a building's or facility's HVAC (heating, ventilation, and air conditioning) system is now operating. This mode shows how the HVAC system is operating to keep the inside climate at the appropriate level.

For example, at night mode is set to economy and then it can be changed to pre-comfort and the comfort. This mode can be configured automatically based on sensor or based on pre-defined configuration.

Room Air Quality (CO2)

This is the amount of carbon dioxide (CO2) in the air in a particular room or interior space, expressed in parts per million (ppm).

Although CO2 is a naturally occurring element of the Earth's atmosphere, human actions like breathing, burning fossil fuels, and poor ventilation can cause concentrations of the gas to rise indoors. To evaluate indoor air quality (IAQ), it is critical to monitor CO2 levels since elevated concentrations may be a sign of inadequate ventilation, which could cause occupant discomfort, weariness, or health problems.

For an appropriate IAQ, CO2 levels in interior areas are generally advised to be between 400 and 1000 ppm, with levels under 1000 ppm being preferable. Nevertheless, requirements and rules might change based on things like building type, occupant density, and local laws.

Room Temperature

The term "room temperature" usually describes the air temperature in specific rooms, workstations, or areas where people operate. It is essential to keep rooms at the proper temperature for the comfort, productivity, and wellbeing of its occupants.

Comfort preferences can differ amongst persons. For normal office environments, the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) recommends a temperature range of 23-26° C.

External weather conditions

The operation, efficiency, and energy consumption of HVAC systems are significantly impacted by external weather conditions.

Temperature: The amount of heat and air conditioning that a structure need depends directly on the outside temperature. Cooling systems are used to lower inside temperatures in warmer weather, and heating systems must work harder in colder weather to keep comfortable indoor temperatures.

Humidity: The performance of HVAC systems and indoor humidity levels can be influenced by outdoor humidity levels. Elevated relative humidity can cause interior spaces to seem hotter and stuffier, necessitating the HVAC system to run longer to cool down. By eliminating extra moisture from the air, HVAC systems with dehumidification capabilities aid in maintaining ideal indoor humidity levels.

Wind: The direction and speed of the wind have an impact on how easily outside air enters buildings through ventilation apertures, gaps, and fractures. To preserve thermal comfort and interior air quality, HVAC systems need to take outdoor air infiltration rates into consideration. Due to their effects on airflow and heat transmission, windy weather can also reduce the efficiency of HVAC equipment, including rooftop units and outdoor condenser coils.

Precipitation: Outdoor HVAC equipment may be impacted by rain, snow, and other types of precipitation. For HVAC systems to continue operating during bad weather and to avoid water damage, proper drainage and protection of exterior units are crucial. Furthermore, power outages, equipment damage, or safety issues can cause HVAC systems to malfunction during extreme weather events like hurricanes or blizzards.

Data collection and manage data.

Oslobygg owns approximately 2000 of buildings in Oslo. These buildings are kindergarten, School, Care building Oslo, Stadiums, Education building, Culture and Sports building. Weather in Oslo is extreme cold during the winter and summer is just for limited period. Maintaining indoor temperature and good air quality is important in different type of rooms. Project engineers shared some lights on data collection. Different type of raw data available from different sensors. In this project data is collected from HVAC systems from one old classroom which one is not recently renovated and one recently renovated classroom. Usually, renovation brings new technology and improved and updated HVAC system.

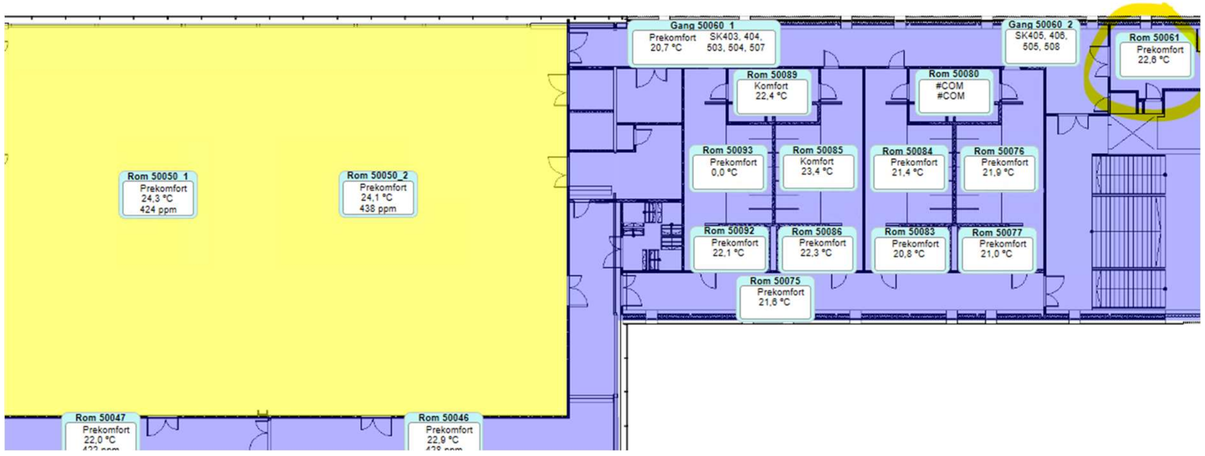
Data collection and building dataset.

Datasets supply the raw data from which algorithms learn patterns, relationships, and correlations, datasets are essential for the development of AI. The quantity and caliber of the dataset frequently have a big impact on how well AI models work and how well they can generalize. Furthermore, datasets need to be carefully selected to guarantee that they are varied, representative, and free of biases that can impair the effectiveness or equity of AI systems.

It was challenging to get correct data from the system. Oslobygg uses different vendors, like Siemens [14], Johnson Controls [15], Schneider Electric [16] and Bravida [17].

Discussion to get correct dataset was a long process. Different approval and cost were involved in part of the process. Internal meetings to choose correct vendor and type of operation of the HVAC done and several external discussions with vendor to know the data access possibilities done.

Merging the different source data into one dataset was challenging. This dataset was built with data from heating radiator, internal and external pump, and sensor data measurement of Co2 in the air. All different data pushed into database and SQL queries and table join queries used to build the final dataset.



Figur 2: Building map to point the room in Hasle school.

Image taken from Oslobygg's building structure diagram form Hasle school and the marked area is a meeting room, and the HVAC system is an old installed system.

We received data from Siemens IOT systems from Hasle School. We choose one old and one recently renovated classroom to do some comparison. We received data from January 2022 we got data round the clock and frequency is every 30 minutes.

It was crucial to know the weather condition of Oslo during the same period. External weather data dump received from "open weather map" to compare the indoor and outdoor situation every half hour since 2022 January. [18]

Data as CSV dump received from vendors and then imported into SQL server database. It was important to categorize the data and plan to level the data and mapping with correct time stamp from all these different raw sources.

Data Modeling and building of data warehouse.

The requirement for simple access to an organized repository of high-quality data that can be utilized for decision-making gave rise to the idea of data warehousing. It is widely acknowledged that information is a very impotent resource that can give any organization major advantages as well as a competitive edge in the business sector. Despite having enormous volumes of data, organizations are finding it more and more difficult to access and utilize it. This is since it is available in a multitude of formats, runs on a wide range of systems, and is housed in an array of file and database structures created by many suppliers. As a result, companies have had to create and manage hundreds of programs to collect, process, and combine data for usage. [19]

The process of accessing disparate data sources, cleaning, filtering, and transforming the data, and storing it in an accessible, understandable, and useful structure is implemented by data warehousing. After then, the data is used for reporting, data analysis, and query. Because of this, the requirements for access, use, technology, and performance are entirely different from those in an operational environment that is transaction oriented. Data warehousing can handle enormous volumes of data, especially when taking the requirements into account.

For the examination of historical data. Large volumes of data must frequently be scanned by data analysis tools, which may have an adverse effect on operational applications because they depend more heavily on performance. To reduce disputes and performance degradation in the operational environment, it is necessary to keep the two environments apart.

There are numerous ways to determine what the needs of a firm are. These techniques often fall into one of two groups: user-driven requirements gathering or source-driven requirements gathering.

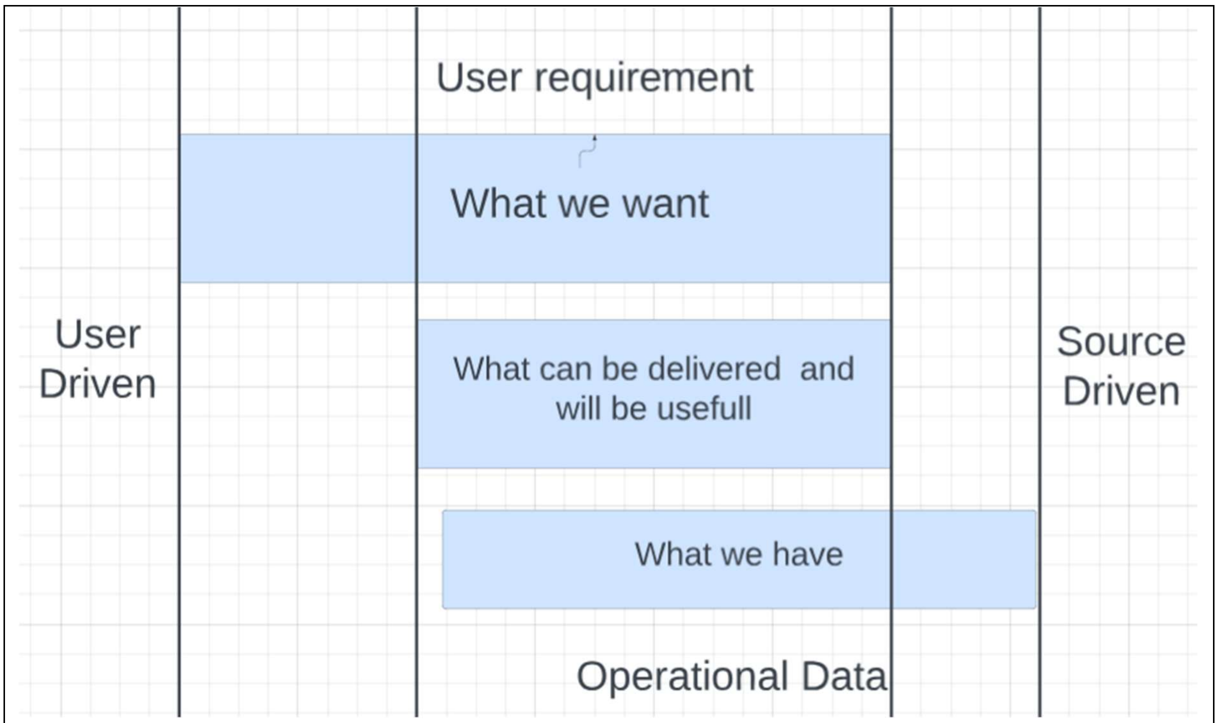


Figure 3: Source-Driven and User-Driven Requirements Gathering

HVAC system consist of different source of data as the sensors can be from different vendors. Data is stored in different format and spread among several tables and sometimes in different source locations. To analyze the overall situation data is essential to get in machine learning readable format. Gathering and formatting the data done with SQL server database. Some random database backup files and export of unstructured data received in very raw format. One common point was considered was time stamp. Since 2022 data collected in every 30 minutes gap for all different components of HVAC systems. Date and time field received as text format some cases. It was first job to convert text to date and time and then converting the date and time in

Data Categorization

It was crucial to set levels and categorize the dataset with correct levels to make the data ready for processing.

DD	**MM**	**YYYY**	**Hour**	**Minute**
20	03	2023	06	30

It was also important to find out season. Winter and summer have different climate situations in Norway.

Month	**Month Name**	**Season id**	**Season**
01	Jan	100	Winter

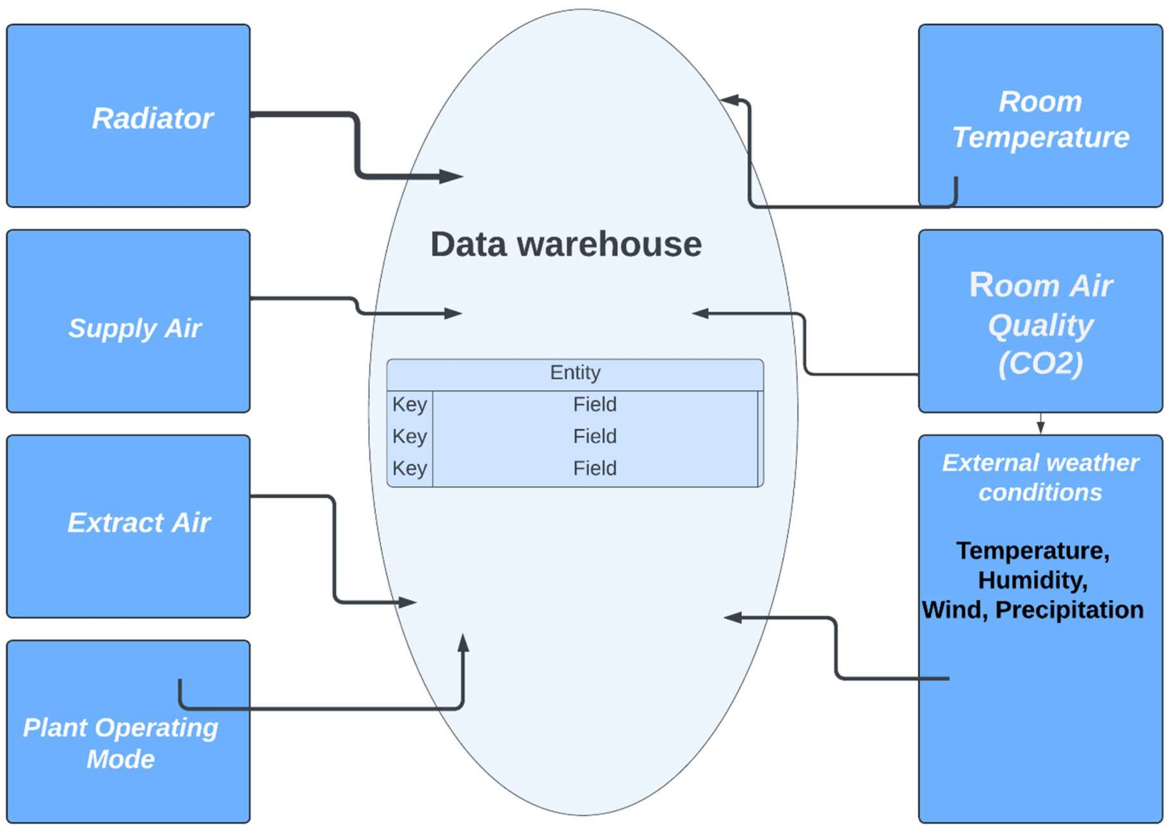
02	July	200	Summer
09	September	300	Fall

HVAC systems are supposed to be configured differently during day and nighttime. It is important to consider the different part of the day. Good to consider that Schools and offices are busy during 08:00 Am to 16:00 and after that it usually different setup for HVAC systems.

Morning	**Afternoon**	**Evening**	**Night**	**Early Morning**
8:00	13:00	7:00	10:00	5:00

Likewise, table was considered to identify different time of the day with ID for dataset.

External weather condition plays essential impact on HVAC system. Some tunings are automatic, and all component adjustments are important to consider in different weather conditions. External weather data set for last few years collected and weather condition like Temperature, humidity, wind, and precipitation were taken into consideration. It also considered every 30 minutes interval and data warehouse made ready to consider all data together with only numeric value to build a dataset.



Figur 4: SQL Server Express edition is used to build the Datawarehouse, this can be used as a prototype for future to get data flow automatically with help of API connection and Extract, transform, and load (ETL)[39]

This process repeats for three different dataset that build for comparison. This data collection and data formatting process was challenging, and finally good and clean datasets made from these.

Dataset details	**Approximate data volume**
Hasle school old meeting room	38000
Hasle school newly renovated classroom	39000
Bryn Seng school newly renovated classroom	30000

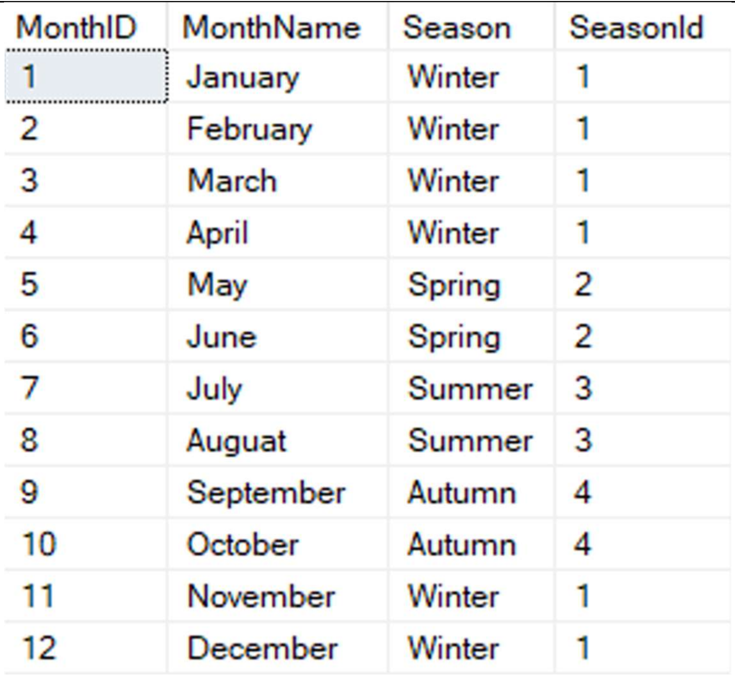
HVAC component table

This table created in the database to identify the text field with Id, those ids used for table sql joining purposes.

Radiator Valve Id	Radiator valve
Sypply Air Id	Supply Air valve
Extract Air id	Extract air valve
Plant operation mode Id	Plant operation mode
Room co2 id	Room Co2 collector
Room temperature Id	Room Temperature collector

Categorization of data with Season

Seasons are very prominent in Oslo and data categorization is important for future analysis.



MonthID	MonthName	Season	SeasonId
1	January	Winter	1
2	February	Winter	1
3	March	Winter	1
4	April	Winter	1
5	May	Spring	2
6	June	Spring	2
7	July	Summer	3
8	Auguat	Summer	3
9	September	Autumn	4
10	October	Autumn	4
11	November	Winter	1
12	December	Winter	1

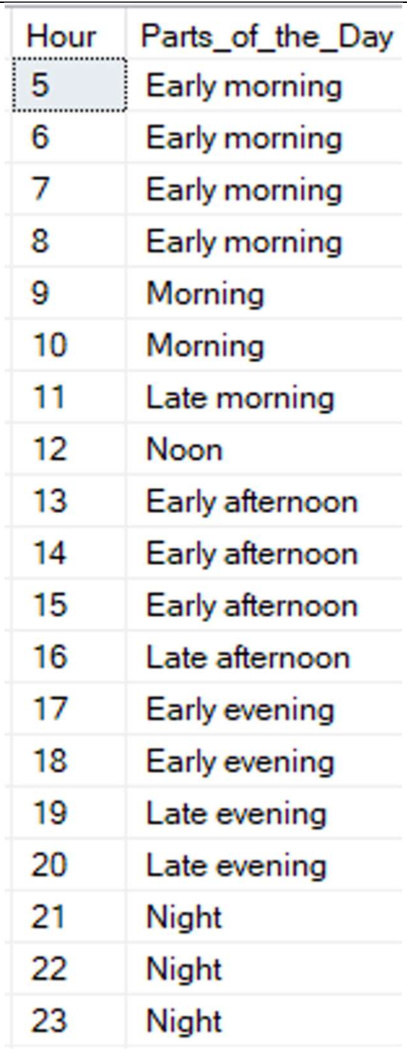
Figure 5: Seasons and month mapping table is database

Various periods of the day

Like in many other nations, Norway divides the day into multiple sections or segments to represent the various periods of the day.

It is important to use the day, time, and month of the year to get know whether it is a day or morning or evening. Because HVAC systems are configured at different mode based on the usage of the place. So, part of the day is very important factor to analyze perfection of the mode of the operation.

School building is mostly used during the daytime and so HVAC operation is working in different mode during different part of the day like morning, day, evening, and night. Part of the day table introduced to map hours with part of the day.



Hour	Parts_of_the_Day
5	Early morning
6	Early morning
7	Early morning
8	Early morning
9	Morning
10	Morning
11	Late morning
12	Noon
13	Early afternoon
14	Early afternoon
15	Early afternoon
16	Late afternoon
17	Early evening
18	Early evening
19	Late evening
20	Late evening
21	Night
22	Night
23	Night

Figur 6: Defining part of the day in sql

Season

Weather varies significantly with the change of the season especially Norwegian winter and summer is very different. Winter outside temperature goes down to -25 and during mid-summer temperature reaches + 34. It is important to consider the season.

Winter (December to February): Winter in Norway is marked by low temperatures, snowfall, and a limited amount of daylight.

Spring (March to May): In Norway, springtime heralds the end of winter and the beginning of milder weather and longer days.

Summer (June to August): In Norway, summer is defined by long days, pleasant to warm temperatures, and the midnight sun phenomena (where the sun is visible for 24 hours every day in the northern regions).

Autumn (September to November): The changing of the seasons in Norway results in reduced temperatures.

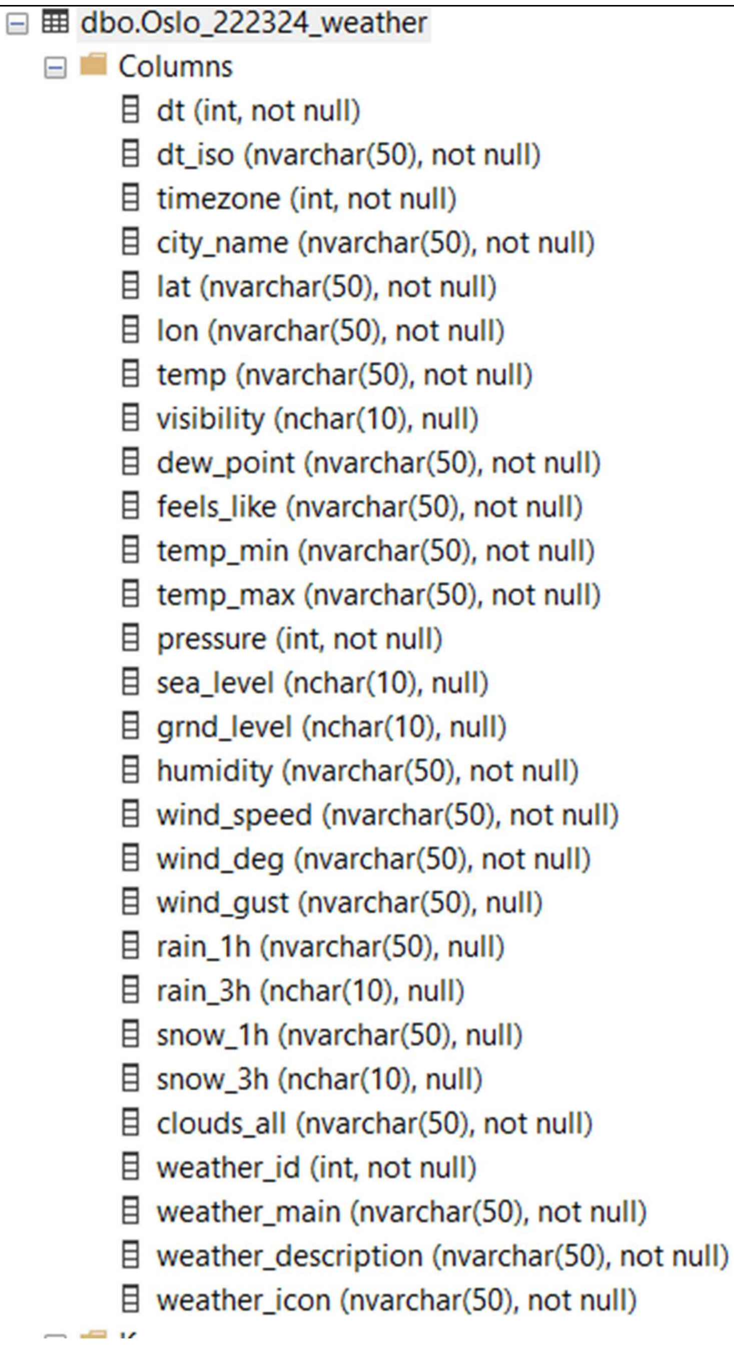
External weather

Norway's wide coastline, diverse topography, and proximity to the North Atlantic Ocean result in a wide variety of meteorological conditions.

The weather in Norway is known for its contrasts and diversity, providing a wide variety of climatic conditions throughout its various regions and landscapes.

Weather information acquired from open weather at open weather map web site. [18]

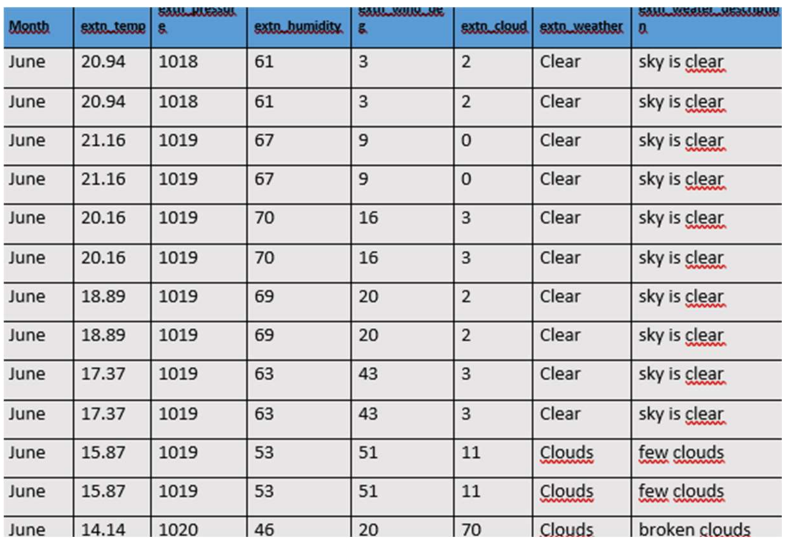
That is the hourly weather in Oslo for the past five years. This makes comparing indoor and outdoor environments easier. Temperature, dew point temperature, pressure, humidity, wind speed, precipitation, cloud cover, and weather description are a few important variables.



```
dbo.Oslo_222324_weather
  Columns
    dt (int, not null)
    dt_iso (nvarchar(50), not null)
    timezone (int, not null)
    city_name (nvarchar(50), not null)
    lat (nvarchar(50), not null)
    lon (nvarchar(50), not null)
    temp (nvarchar(50), not null)
    visibility (nchar(10), null)
    dew_point (nvarchar(50), not null)
    feels_like (nvarchar(50), not null)
    temp_min (nvarchar(50), not null)
    temp_max (nvarchar(50), not null)
    pressure (int, not null)
    sea_level (nchar(10), null)
    grnd_level (nchar(10), null)
    humidity (nvarchar(50), not null)
    wind_speed (nvarchar(50), not null)
    wind_deg (nvarchar(50), not null)
    wind_gust (nvarchar(50), null)
    rain_1h (nvarchar(50), null)
    rain_3h (nchar(10), null)
    snow_1h (nvarchar(50), null)
    snow_3h (nchar(10), null)
    clouds_all (nvarchar(50), not null)
    weather_id (int, not null)
    weather_main (nvarchar(50), not null)
    weather_description (nvarchar(50), not null)
    weather_icon (nvarchar(50), not null)
```

Figur 7: Database Columns from weather table

Sample external weather data represented in image below.

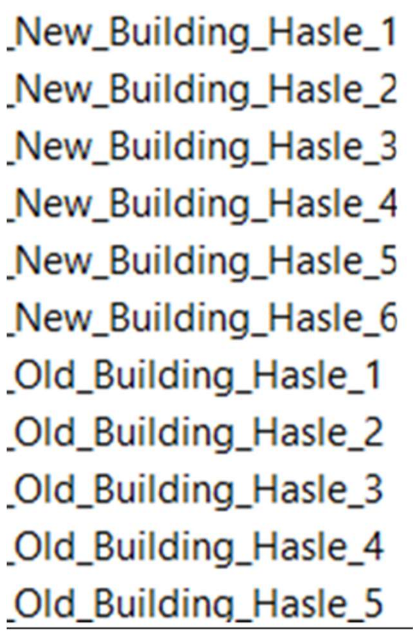


Month	extn_temp	extn_pressure	extn_humidity	extn_wind_deg	extn_cloud	extn_weather	extn_weater_description
June	20.94	1018	61	3	2	Clear	sky is clear
June	20.94	1018	61	3	2	Clear	sky is clear
June	21.16	1019	67	9	0	Clear	sky is clear
June	21.16	1019	67	9	0	Clear	sky is clear
June	20.16	1019	70	16	3	Clear	sky is clear
June	20.16	1019	70	16	3	Clear	sky is clear
June	18.89	1019	69	20	2	Clear	sky is clear
June	18.89	1019	69	20	2	Clear	sky is clear
June	17.37	1019	63	43	3	Clear	sky is clear
June	17.37	1019	63	43	3	Clear	sky is clear
June	15.87	1019	53	51	11	Clouds	few clouds
June	15.87	1019	53	51	11	Clouds	few clouds
June	14.14	1020	46	20	70	Clouds	broken clouds

Figur 8: External weather details

HVAC data inserted into SQL Server table.

There are six different types of data (Radiator valve, Supply Air, Extract air, Plant operation mode, Room Co2, Room Temperature) coming from different IOT components. Data stored in the database in a structured way to use them or merge them as per the requirement.

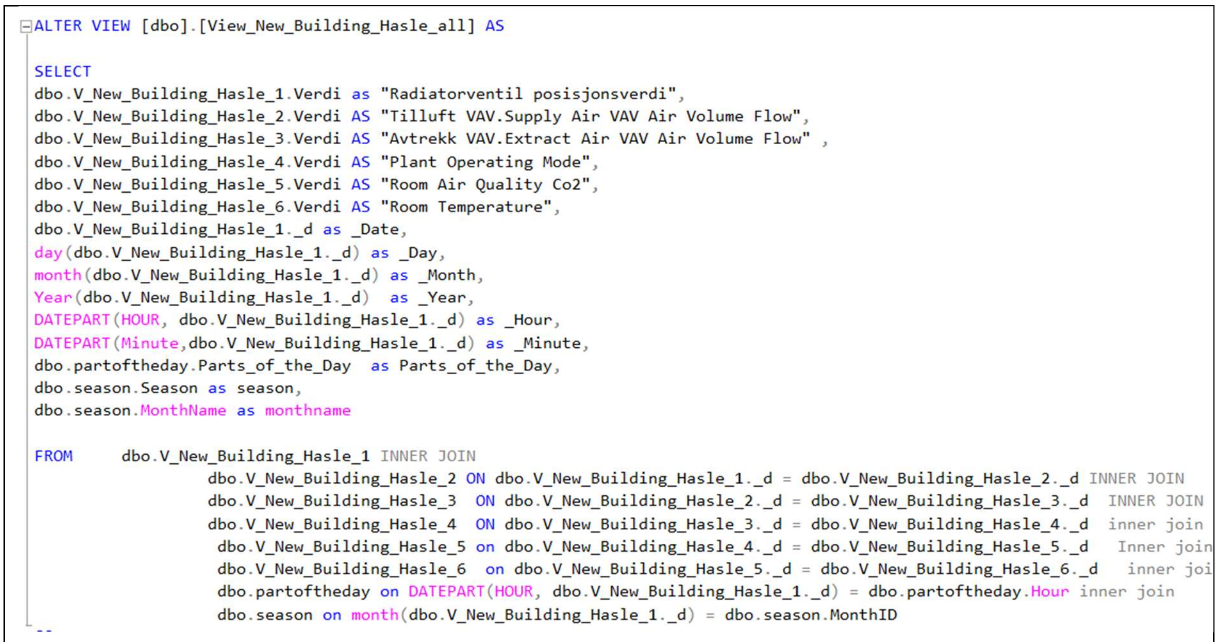


```
_New_Building_Hasle_1
_New_Building_Hasle_2
_New_Building_Hasle_3
_New_Building_Hasle_4
_New_Building_Hasle_5
_New_Building_Hasle_6
_Old_Building_Hasle_1
_Old_Building_Hasle_2
_Old_Building_Hasle_3
_Old_Building_Hasle_4
_Old_Building_Hasle_5
```

Figur 9: Database for Hasle school contains six tables for 6 different components per room.

SQL Joining used to build the base of the dataset

All the data for sources are gathered based on date and time. Day, month, year hour and minute considered and keys to join different tables. Also, part of the day, season table merged with the same dataset. It resulted a consolidated result and one view is created.

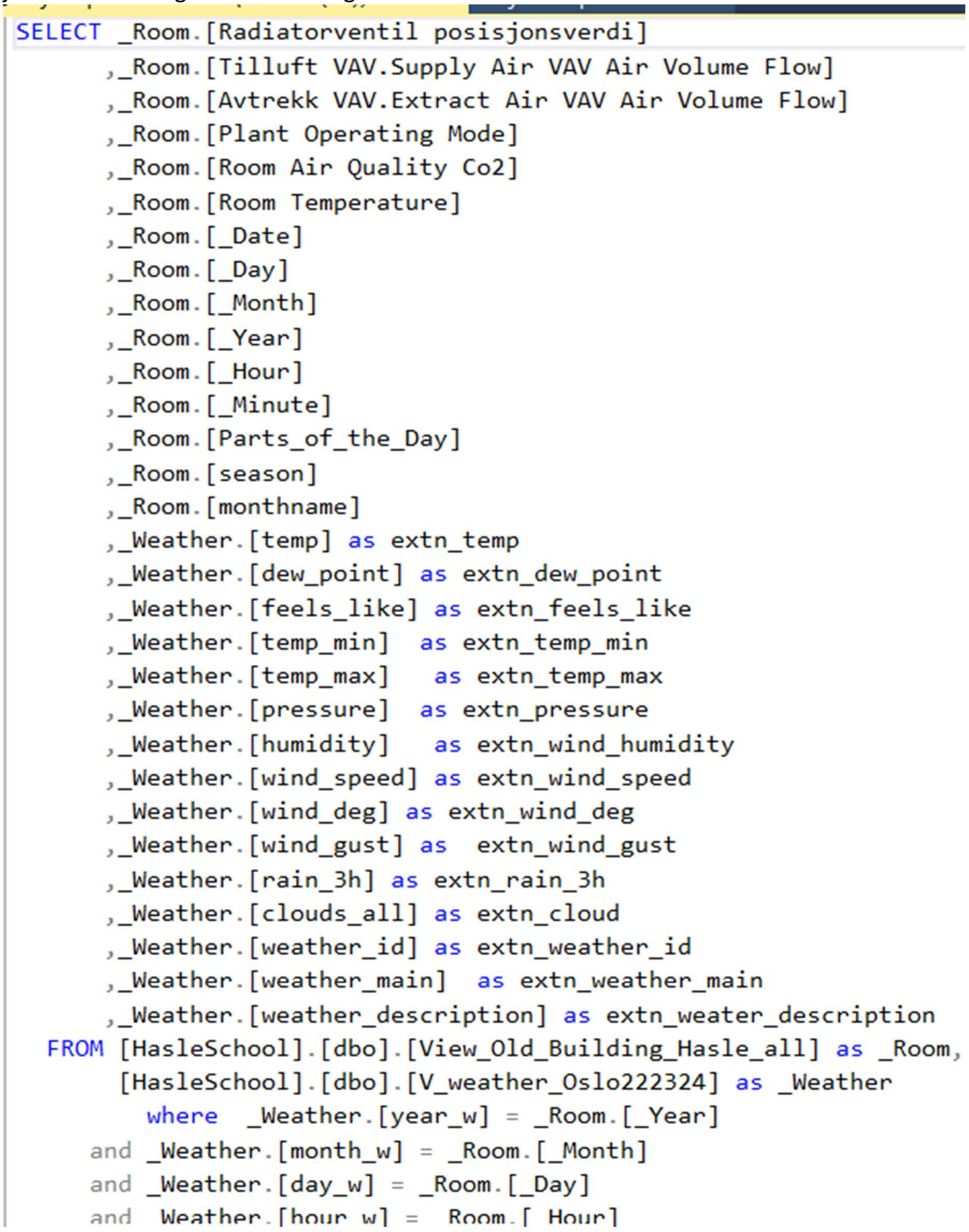


```sql
ALTER VIEW [dbo].[View_New_Building_Hasle_all] AS

SELECT
dbo.V_New_Building_Hasle_1.Verdi as "Radiatorventil posisjonsverdi",
dbo.V_New_Building_Hasle_2.Verdi AS "Tilluft VAV.Supply Air VAV Air Volume Flow",
dbo.V_New_Building_Hasle_3.Verdi AS "Avtrekk VAV.Extract Air VAV Air Volume Flow" ,
dbo.V_New_Building_Hasle_4.Verdi AS "Plant Operating Mode",
dbo.V_New_Building_Hasle_5.Verdi AS "Room Air Quality Co2",
dbo.V_New_Building_Hasle_6.Verdi AS "Room Temperature",
dbo.V_New_Building_Hasle_1._d as _Date,
day(dbo.V_New_Building_Hasle_1._d) as _Day,
month(dbo.V_New_Building_Hasle_1._d) as _Month,
Year(dbo.V_New_Building_Hasle_1._d)  as _Year,
DATEPART(HOUR, dbo.V_New_Building_Hasle_1._d) as _Hour,
DATEPART(Minute,dbo.V_New_Building_Hasle_1._d) as _Minute,
dbo.partoftheday.Parts_of_the_Day  as Parts_of_the_Day,
dbo.season.Season as season,
dbo.season.MonthName as monthname

FROM     dbo.V_New_Building_Hasle_1 INNER JOIN
                 dbo.V_New_Building_Hasle_2 ON dbo.V_New_Building_Hasle_1._d = dbo.V_New_Building_Hasle_2._d INNER JOIN
                 dbo.V_New_Building_Hasle_3  ON dbo.V_New_Building_Hasle_2._d = dbo.V_New_Building_Hasle_3._d  INNER JOIN
                 dbo.V_New_Building_Hasle_4  ON dbo.V_New_Building_Hasle_3._d = dbo.V_New_Building_Hasle_4._d  inner join
                  dbo.V_New_Building_Hasle_5 on dbo.V_New_Building_Hasle_4._d = dbo.V_New_Building_Hasle_5._d   Inner join
                  dbo.V_New_Building_Hasle_6  on dbo.V_New_Building_Hasle_5._d = dbo.V_New_Building_Hasle_6._d   inner joi
                  dbo.partoftheday on DATEPART(HOUR, dbo.V_New_Building_Hasle_1._d) = dbo.partoftheday.Hour inner join
                  dbo.season on month(dbo.V_New_Building_Hasle_1._d) = dbo.season.MonthID
--
```

Figur 10: For each room one common view created consolidating all HVAC sensor values.

Merger of internal and external weather with SQL join

There is already one view from room HVAC system and table from external weather. SQL join used to merge these two to get the final dataset.



```
SELECT _Room.[Radiatorventil posisjonsverdi]
      ,_Room.[Tilluft VAV.Supply Air VAV Air Volume Flow]
      ,_Room.[Avtrekk VAV.Extract Air VAV Air Volume Flow]
      ,_Room.[Plant Operating Mode]
      ,_Room.[Room Air Quality Co2]
      ,_Room.[Room Temperature]
      ,_Room.[_Date]
      ,_Room.[_Day]
      ,_Room.[_Month]
      ,_Room.[_Year]
      ,_Room.[_Hour]
      ,_Room.[_Minute]
      ,_Room.[Parts_of_the_Day]
      ,_Room.[season]
      ,_Room.[monthname]
      ,_Weather.[temp] as extn_temp
      ,_Weather.[dew_point] as extn_dew_point
      ,_Weather.[feels_like] as extn_feels_like
      ,_Weather.[temp_min]  as extn_temp_min
      ,_Weather.[temp_max]   as extn_temp_max
      ,_Weather.[pressure]  as extn_pressure
      ,_Weather.[humidity]   as extn_wind_humidity
      ,_Weather.[wind_speed] as extn_wind_speed
      ,_Weather.[wind_deg] as extn_wind_deg
      ,_Weather.[wind_gust] as  extn_wind_gust
      ,_Weather.[rain_3h] as extn_rain_3h
      ,_Weather.[clouds_all] as extn_cloud
      ,_Weather.[weather_id] as extn_weather_id
      ,_Weather.[weather_main]  as extn_weather_main
      ,_Weather.[weather_description] as extn_weater_description
  FROM [HasleSchool].[dbo].[View_Old_Building_Hasle_all] as _Room,
      [HasleSchool].[dbo].[V_weather_Oslo222324] as _Weather
        where  _Weather.[year_w] = _Room.[_Year]
     and _Weather.[month_w] = _Room.[_Month]
     and _Weather.[day_w] = _Room.[_Day]
     and _Weather.[hour_w] = _Room.[_Hour]
```

Figur 11:This is the old room dataset ready with combination HVAC and external weather time series data.

Discussion on Datawarehouse

This is the concept for getting data directly from the IOT systems. Microsoft introduces the concept of IOT Hub [20]. This concept Datawarehouse system can be designed in a most robust way and possible to connect to IOT data sources different vendors. This case old historical data sample exported and used but using application programming interface (API) [21]. Data consumption can be automatic with ETL [22] tools.

Project data sample connection and database design shows how to use database in a simple way to deal the massive amount of data and train the system, predict, and check the overall HVAC performance and failures. This leads to several performance and energy savings in future.

Methods, concepts to pre-process the data.

Data collection and data validation was very important. Received data formatting and fixing exceptions were essential for AI.

Feature engineering:

The act of choosing, modifying, and converting unprocessed data into features that may be applied to supervised learning is known as feature engineering.

Feature creation: Developing new variables that will be most beneficial to our model is the process of building features.

Transformations: A function that only converts features between different representations is known as feature transformation. Here, plotting, and visualizing data is the aim.

Feature extraction: The technique of removing features from a data set to find relevant information is known as feature extraction.

Exploratory data analysis (EDA): By examining the qualities of your data, EDA is a straightforward and effective approach that can help you better understand it.

Benchmark: The most approachable, trustworthy, transparent, and comprehensible model that you can compare your own to is known as a benchmark.

Handling all the missing values:

Data must be consistent and important to deal the null values with relevant action. For example, setting zero for all missing values.

Understanding the correlation

The statistical summary of the relationship between two sets of variables is called a correlation. It is an essential component of many sophisticated machine learning approaches as well as a fundamental component of data exploratory analysis.

One method to demonstrate the potential connections between the values of several variables is correlation. The adage "correlation does not imply causation" refers to the fact that a correlation's measure alone does not establish causality. A thorough grasp of the relationship between the variables and an adequate number of data points are prerequisites for evaluating whether correlations could be helpful in making predictions.

A least-squares measure of the error between an estimating line and the actual data values, normalized by the square root of their variances, is used to compute the correlation coefficient, also known as Pearson's. Perfect direct correlation is represented by a coefficient of 1, whereas perfect inverse correlation is represented by a value of -1. Zero indicates no association.

Correlation Heatmap

An indispensable tool for exploratory data analysis is the correlation matrix. The same information is presented in a visually appealing manner in correlation heatmaps. Furthermore, they alert us to potential multicollinearity issues and quickly display which variables are associated, to what extent, and in which direction.

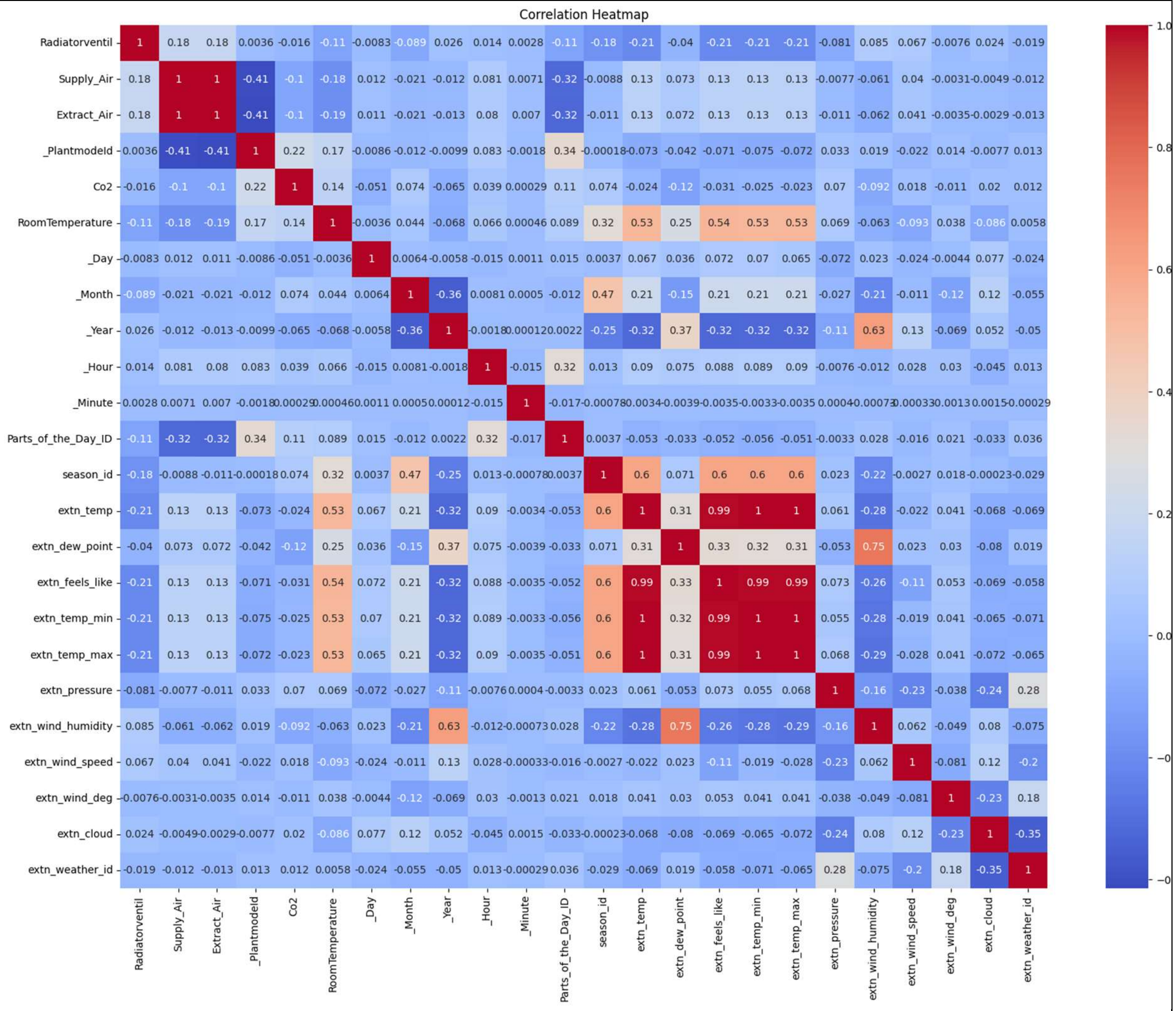
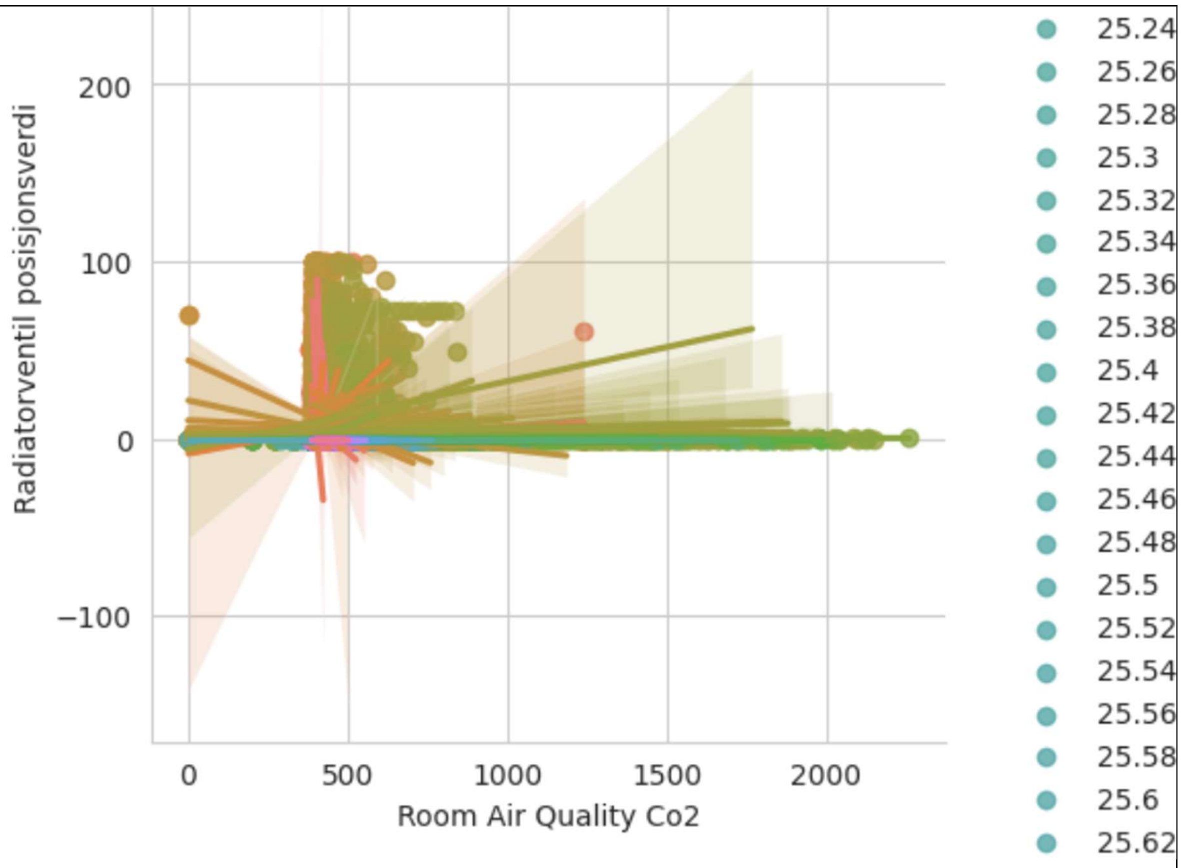


Figure 12: Heatmap

It's visible that the supply air, extract air, external temperature, external humidity are very much corelated with each other. Stronger relationships are shown by darker hues, and weaker correlations by lighter hues. Warm hues like red or orange are typically used to symbolize positive correlations, which occur when one variable tends to grow as the other does.
Cool hues like blue or green are typically used to depict negative correlations, which occur when one variable tends to increase while the other tends to decrease.

Regression plots

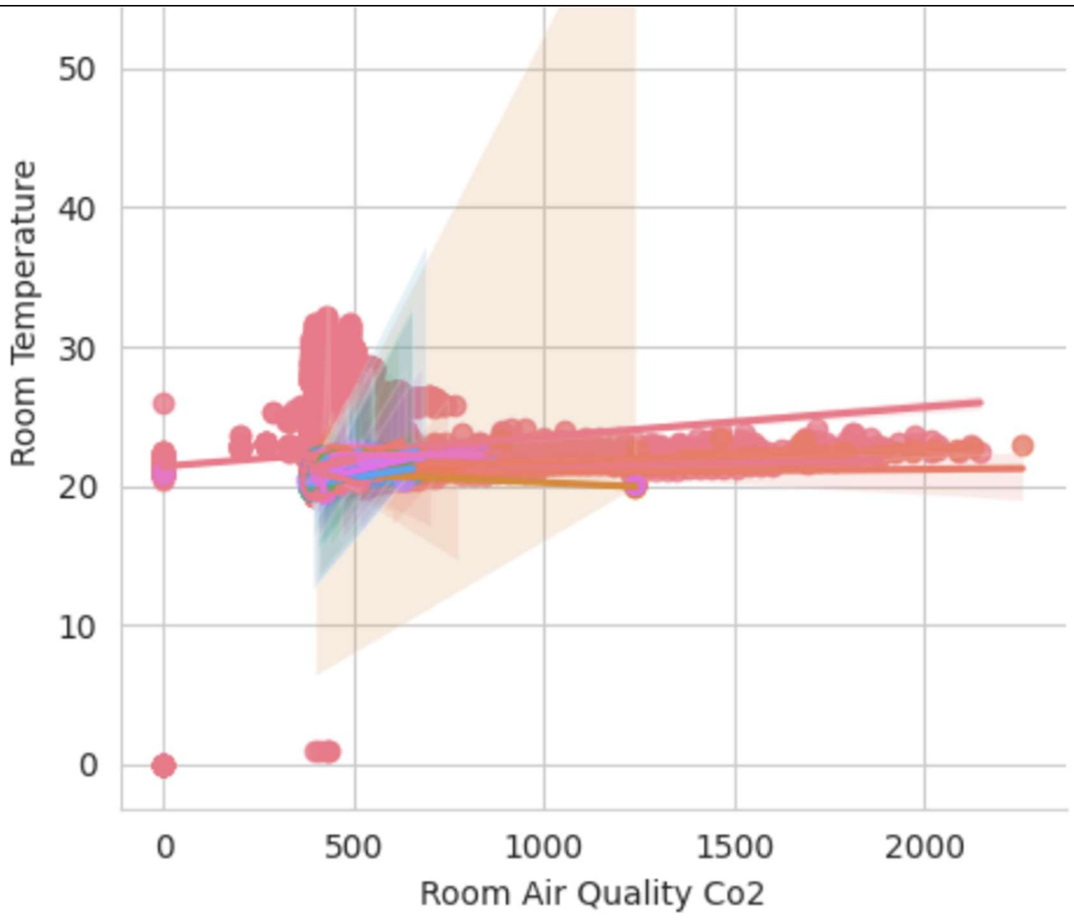
When conducting exploratory data analyses, a visual guide can highlight trends in a dataset. To depict the linear correlations between two parameters, regression plots, as their name implies, draw a regression line between them.



Figur 13: Regression plot, radiation and Co2 with Temperature

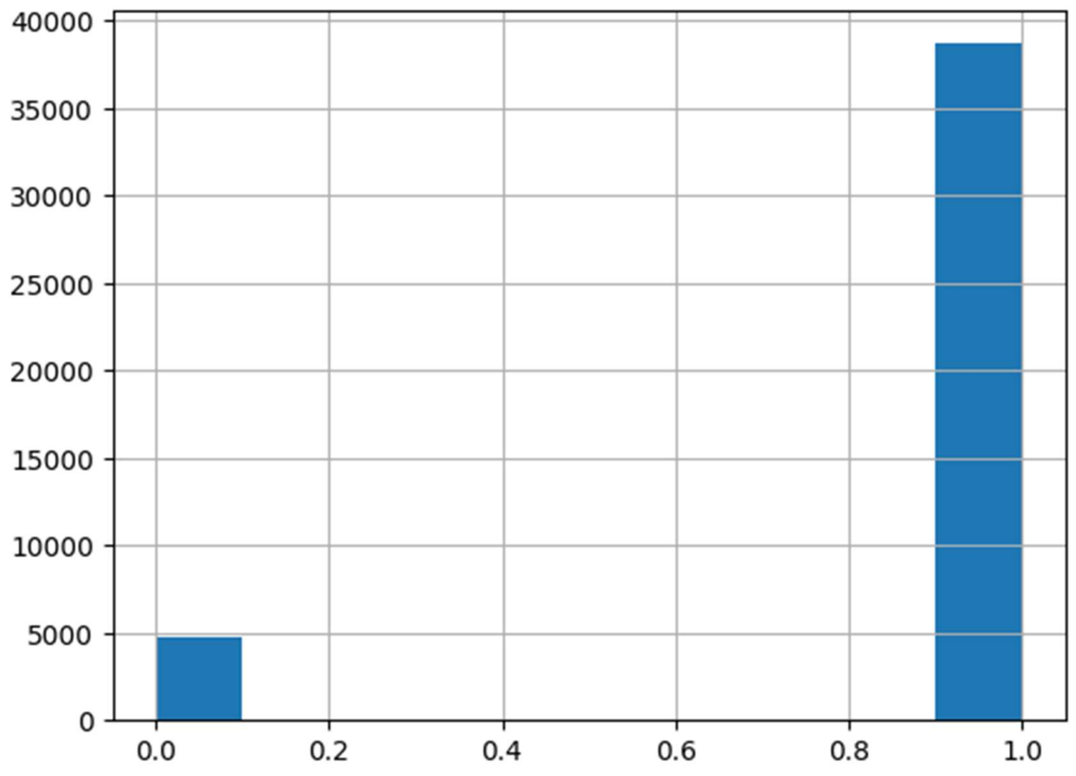
This is the analysis between heating control and Co2 level inside the school room, where room temperature is part of the analysis, we see the impact analysis.

The image bellow shows the relation of room air quality and room temperature.



Figur 14: Regression plot, Co2 and room Temperature

Let us consider good quality <u>Co2 is (<499 PPM)</u> for small kids. Random Forest Regressor can be used to determine the feature and target feature.



Figur 15: Random Forest Regressor, co2> 499

A random forest is a meta estimator that employs averaging to increase prediction accuracy and manage over-fitting. It fits multiple decision tree regressors on different sub-samples of the dataset.

A non-parametric supervised learning approach that is used for both regression and classification applications is the decision tree. With a root node, branches, internal nodes, and leaf nodes, it has a hierarchical tree structure.

Trying out different models and finding scores

Logistic Regression: 99.31234337380556

Decision Tree: 100.0

Ridge Regression: 44.9420443307315

Lasso Regression: 44.66871575755368

Random Forest: 100.0

Gradient Boosting: 100.0

Support Vector Machine: 99.25714928399529

K-Nearest Neighbors: 99.14675290647337

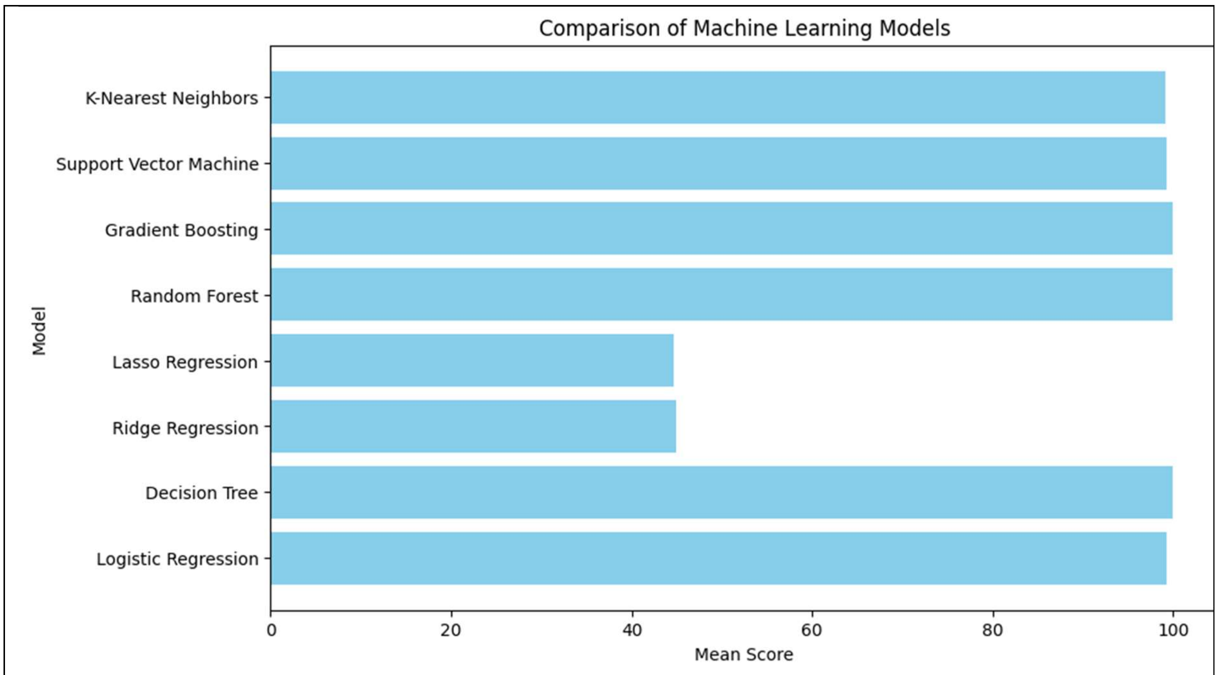


Figure 16: Machine Learning Models

Classification models with their score after trying with Decision Tree Classifier and Random Forest Classifier.

Decision Tree: 1.0

Random Forest: 1.0

Once the data has been cleaned, preprocessed, and organized, the first thing we do is input it into a superior model, from which we naturally obtain output in probabilities. To get improved performance and greater effectiveness, and that is just what we desire. This is the point at which the Confusion matrix becomes visible. A performance metric for machine learning classification is the confusion matrix.

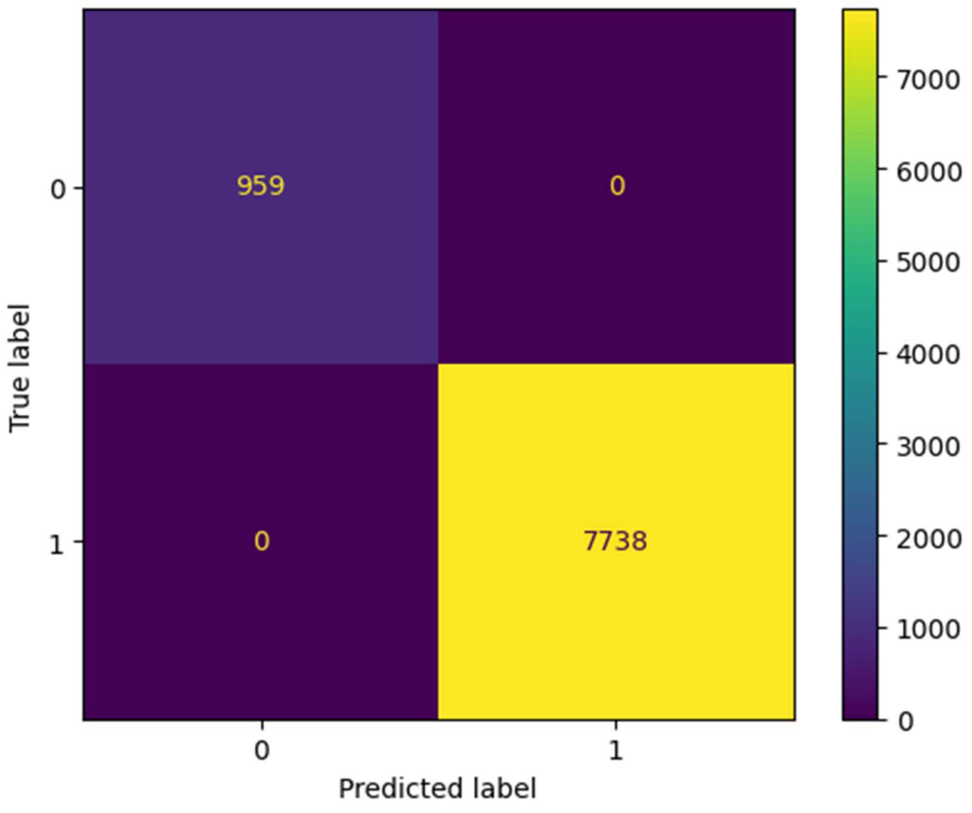


Figure 17:A performance metric for machine learning classification

Feature Generation

The time-series operational data that are logged into the database are not evenly spaced. The observation time is not consistent in a time series data that is irregularly spaced or irregular in timing. Stated differently, all system's observations are recorded at a distinct moment. Because diverse systems do not share a standard logging timeline, it is difficult to identify common patterns among them. The data must be defined on a similar chronology to compare various systems and the recorded system's behavior.

During the feature generation process, it is important to eliminate time-series data that is excessive or poorly spaced and use the raw data to create useful features. A feature is generally defined as the whole amount (of some parameter) per unit. In this instance, a feature is the average value of a chosen parameter daily. To determine the true behavior of the sensor, we take the mean of the observed sensor during a specific time of day. As a result, time-series data with equal spacing is produced, which can be compared to all other systems for additional analysis.

Data Cleaning and Transformation

This stage aims to address the issue of missing data. The data travels via several communication bridges from an HVAC system to the database. Missing data in the database results from a connectivity failure in any one of these bridges. One may experience a few-minute, full-day, or even month-long gap in time. Systems that exhibit absent periods longer than three days in a row are deemed unhealthy.

The unstructured way the sensor data are kept in the back-end database prevents them from being directly utilized for analysis. The sensor values are then extracted against a distinct time stamp and converted into a structured CSV format for this case.

Prediction in machine learning, or machine learning prediction, is the result of an algorithm trained on a historical dataset. Then, for every record of the fresh data, the algorithm produces likely values for the unknown variables. Projecting a likely data set that connects to the original data is the goal of prediction in machine learning. This aids businesses in forecasting client behavior and industry shifts. Prediction is essentially used to match a shape to the data as nearly as feasible.

In conclusion, interior temperature in Norway still has a big impact on indoor CO2 concentration because of how it interacts with HVAC systems. Effective ventilation techniques, energy-efficient HVAC systems, heat recovery technology, and occupant behavior consideration are crucial for controlling indoor air quality and guaranteeing healthy indoor environments, even in this chilly climate.

Image 20 shows the Co2 levels throughout the day. Co2 < 800 PPM is acceptable and image points during the day school room is fresh and HVAC is operating as it should be, during night or late evening Co2 level increases may be due to some other components are auto stopped by then.

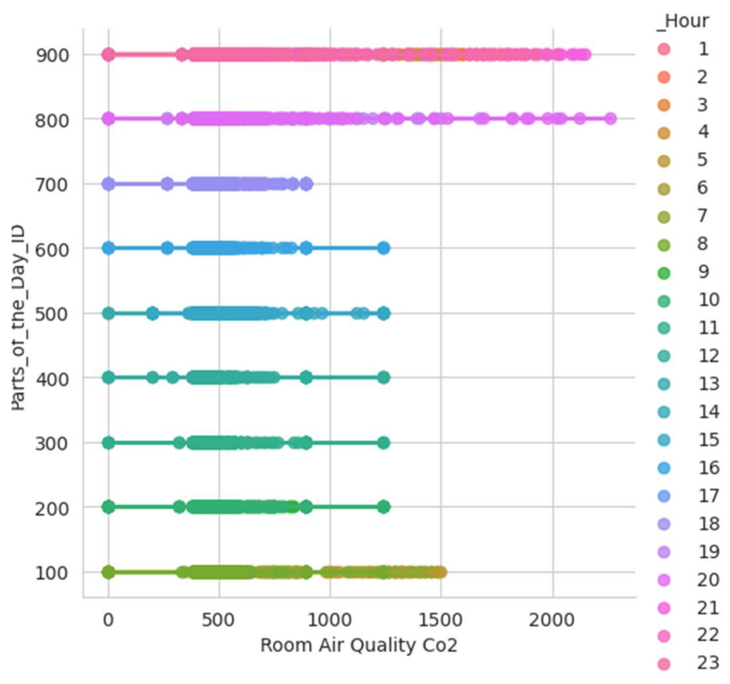


Figure 18:Co2 levers hourly report in classroom

Identify anomalies from classroom data.

Anomaly detection is becoming a critical tool for protecting data integrity and revealing hidden insights. Basically, an anomaly is something that is out of the ordinary, deviates from the norm, or stands out due to its uniqueness within a particular scenario or category. Deviations, sometimes known as anomalies or outliers, may point to the presence of underlying problems, errors, or undesirable behaviors that call for additional investigation. The process of finding data points that, when compared to the bulk of the data in a dataset, are noticeably different, or "outliers," is known as anomaly detection. (Varun Chandola 2009)

Anomalies come in different forms, including contextual, point, and collective anomalies. Individual data points that show notable differences from the rest of the data are called point anomalies. Data points that demonstrate anomalous behavior within a particular circumstance or setting are known as contextual anomalies. When examined separately, though, they might not be regarded as outliers. When a group of connected data points are analyzed collectively, they reveal unusual patterns even though the individual data points may not seem odd.

Time series data points

A collection of data points that are captured at regular intervals is referred to as a time series.

It is crucial to understand their sequence. Each individual data point in the series represents a specific variable of interest's numerical value at a given point in time. These intervals' constancy, which can range from yearly to quarterly, monthly, daily, or even in microseconds, depending on the type of data, is what makes them unique. These kinds of data can be used with time series analysis, a potent technique that reveals underlying patterns, trends, and seasonal variations.

Oslobygg uses HVAC system to maintain the buildings usable and there comes millions of time series data points from different sensors. One point is to consider keeping the rooms cozy and fresh with good quality air circulation. On the other hand, it’s important to consider the cost and loss and failures with time. To guarantee the dependability and accessibility of these vital assets, an asset management framework employing highly qualified reliability engineers is in place. As a result, the capacity to anticipate anomalies and reduce risks is extremely valuable. It also helps to avoid unscheduled downtime and needless maintenance (condition-based versus mandatory maintenance), and it will make it possible to manage these assets' critical components more effectively.

Some popular algorithms

K-Nearest Neighbors, One-Class Support Vector Machines, Isolation Forest, Local Outlier Factor, Artificial Neural Network, Autoencoders

Experiment done with some of them.

Model

The time-series operational data that are logged into the database are not evenly spaced. The observation time is not consistent in a time series data that is irregularly spaced or irregular in timing. Stated differently, all system's observations are recorded at a distinct moment. Because diverse systems do not share a standard logging timeline, it is difficult to identify common patterns among them.

The data must be defined on a similar chronology to compare various systems and the recorded system's behavior. It is crucial to remove extraneous and irregularly spaced time-series data during the feature synthesis process to produce useful features from the raw data.

A feature is generally defined as the whole amount (of some parameter) per unit. In this instance, a feature is the average value of a chosen parameter daily. To determine the true behavior of the sensor, take the mean of the observed sensor during a specific time of day. As a result, time-series data with equal spacing is produced, which can be compared to all other systems for additional analysis.

[23]

Some general steps to follow

Eliminate unnecessary columns
Eliminate duplicates
Managing missing values
Transform data types to their appropriate format.
Exploratory Data Analysis (EDA) was used to represent the Co2 that is over 999 PPM

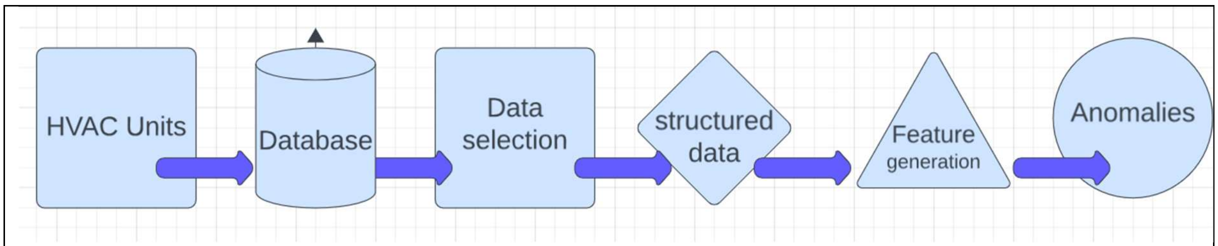
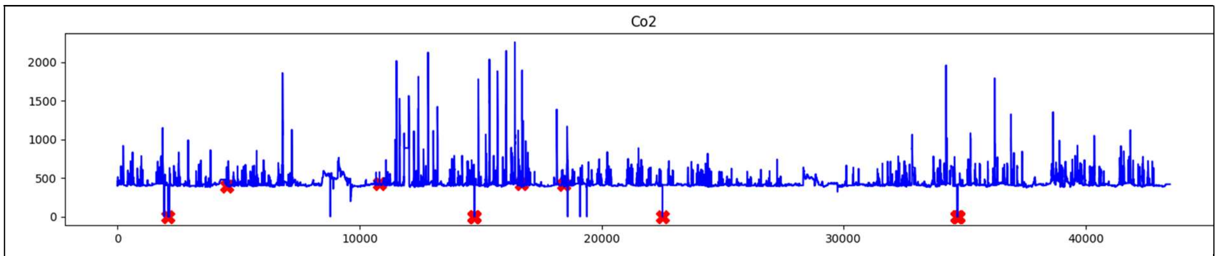


Figure 19: Process to find Anomalies.

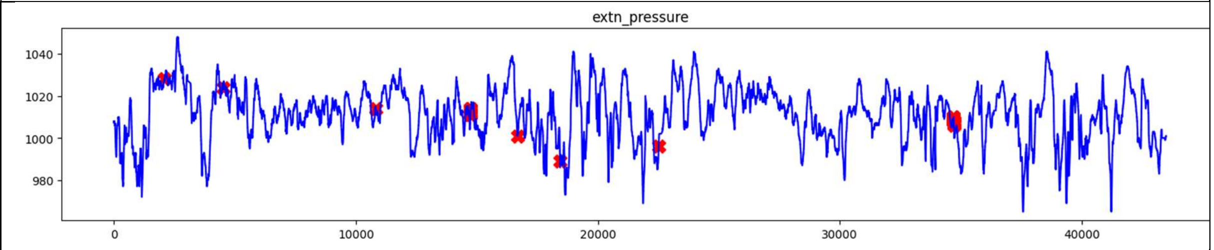
Results

Exploratory Data Analysis

EDA, is an essential first step. To comprehend data's essential features, find patterns, and determine the relationships between variables, it entails analyzing and visualizing the data. refers to the process of examining and analyzing record sets to recognize patterns, find outliers, and determine the correlations between variables.



Figur 20: EDA correlation with Co2



Figur 21: EDA correlation with external air pressure

Autoencoder:

The objective of autoencoders, a subclass of unsupervised neural networks, is to compress input data and subsequently reconstruct it from the compressed representation. The goal of the network's architecture is to reduce the discrepancy between the input and the output that has been rebuilt. [24]

The process was fast to produce anomalies with Co2 (PPM) levels in school room.

Precision: 1.0

Recall: 1.0

F1 Score: (1.0, 1.0, 1.0, None)

Anomaly scores:

```
0       171334.218750
1       171263.265625
2       171300.906250
3       171195.281250
4       171232.609375
             ...
43476   171380.359375
43477   171380.953125
```

```
43478    171343.265625
43479    171300.718750
43480    171263.046875
```

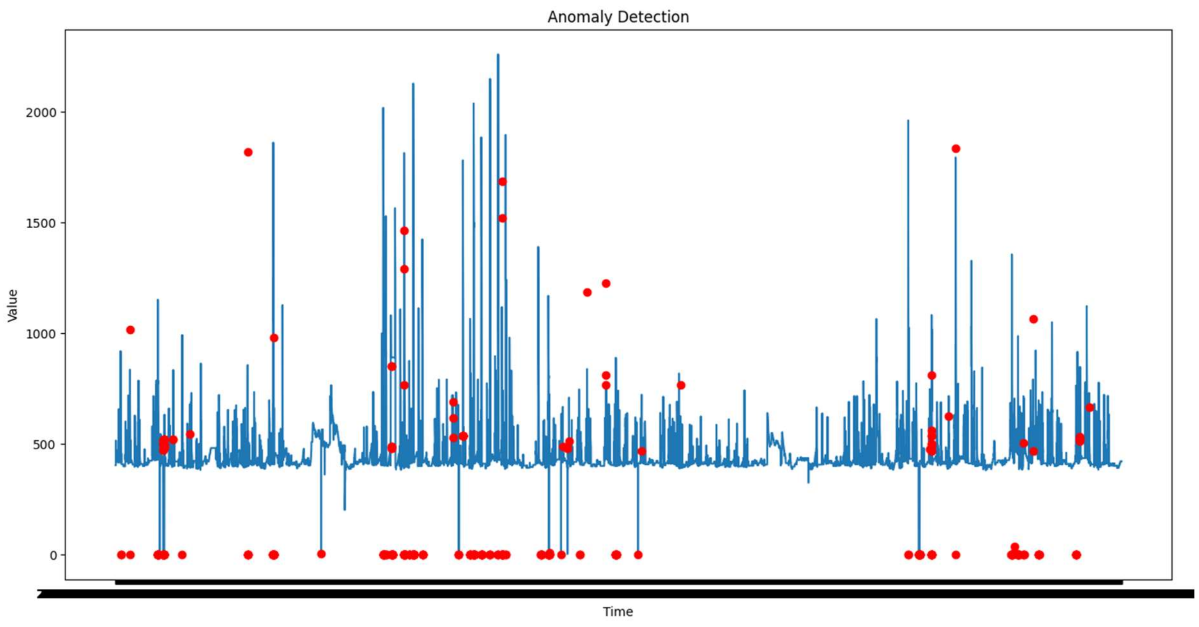


Figure 22: Autoencoder Anomaly detection on time series data

Isolation Forest

With its unique approach to outlier detection, Isolation Forest can quickly identify anomalies. It is among the finest at handling large volumes of data because of its linear time complexity.

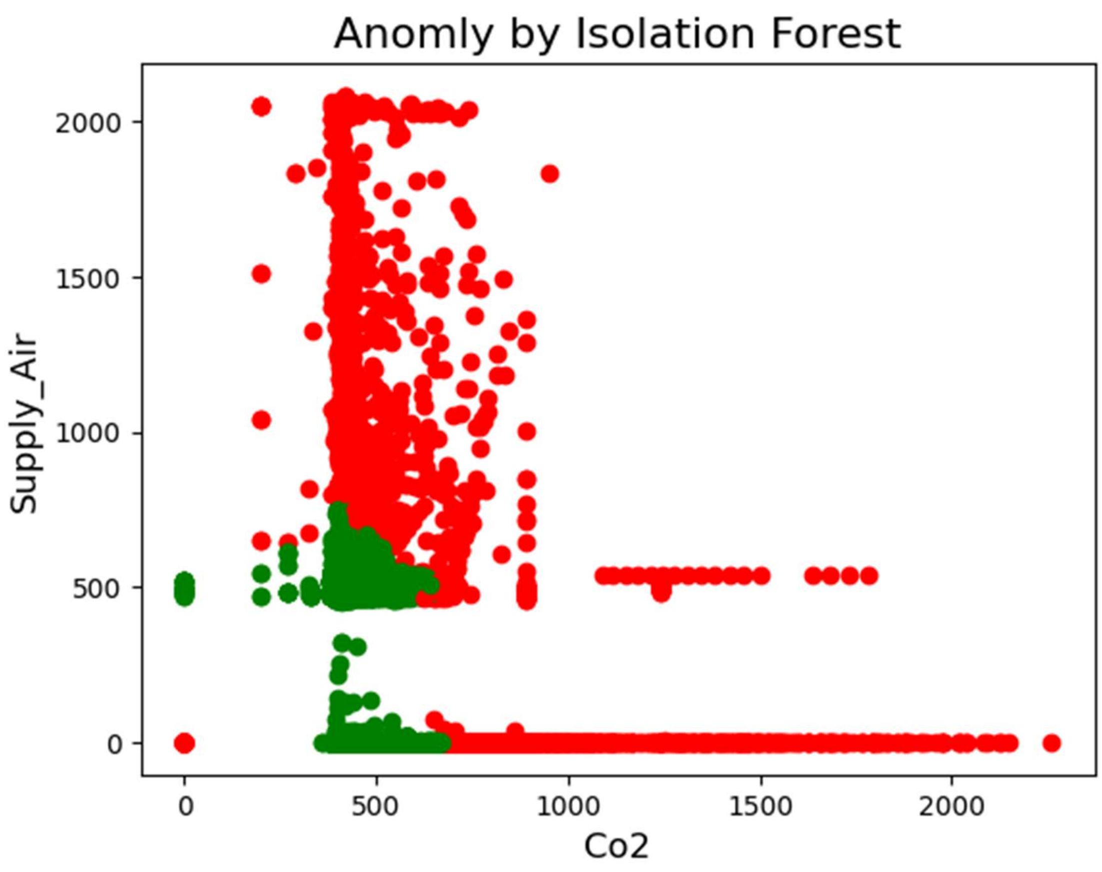


Figure 23: Co2 Anomaly detection with supply air

One-class Support vector machine

SVMs typically seek to generalize based on some input data. They can use this information to determine if any further data are outliers or not. In this case it looks longer time to evaluate

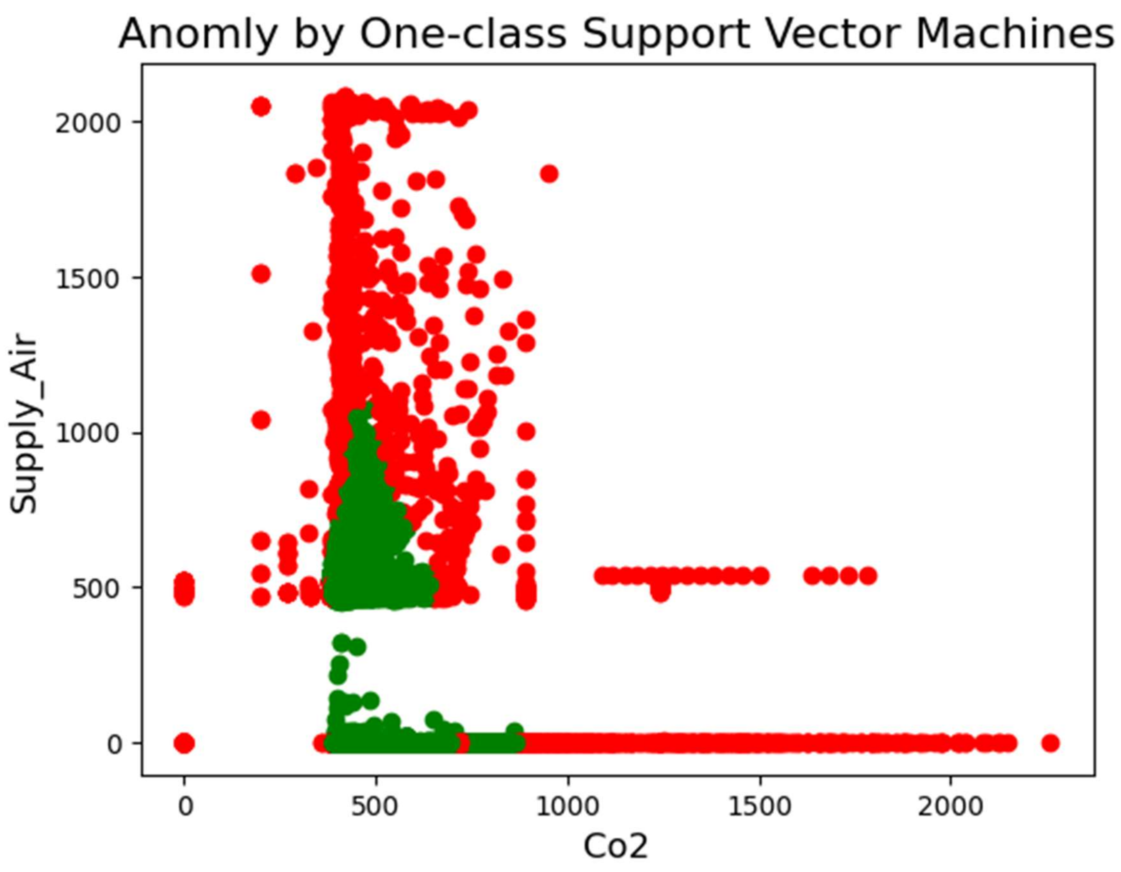


Figure 24: Supply air and Co2

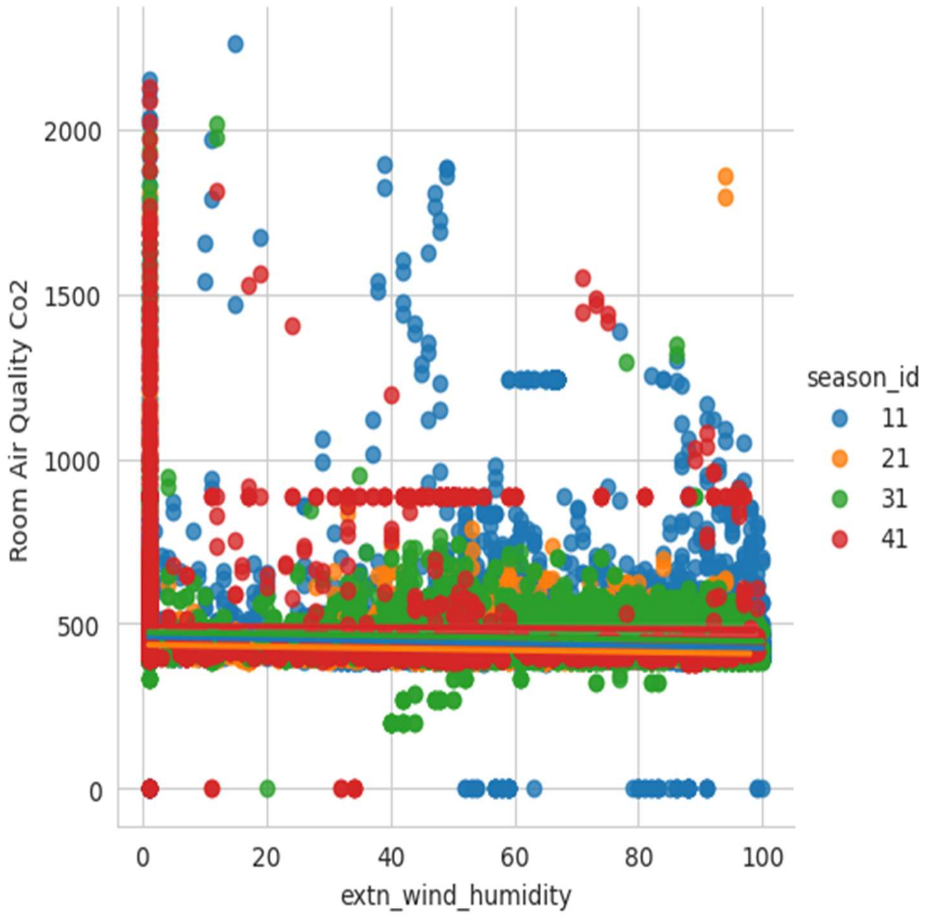
Impact analysis on Indoor CO2 (PPM) levels

Building heating and cooling systems are influenced by temperature. Buildings may be sealed tightly to save energy and keep comfortable inside temperatures during periods of extreme cold or heat. This may result in less air exchange between indoor and outdoor spaces, which could raise CO2 levels indoors if ventilation is inadequate.

HVAC system uses **Economy, pre- comfort, and comfort** mode to make adjustments with these various situations.

Comfort and indoor air quality can be affected by humidity levels. Buildings may need more air conditioning in humid weather to remove moisture from the air, which may have an impact on ventilation rates. On the other hand, humidification systems might be employed to preserve indoor comfort during dry weather, which might have an impact on ventilation.

1,000–2,000 ppm: this level is linked to reports of fatigue and bad air quality. 2,000–5,000 ppm: this level is linked to stuffy, stale, and stagnant air as well as headaches and drowsiness. There may also be a minor feeling of nausea, elevated heart rate, loss of attention, and poor concentration.[25]

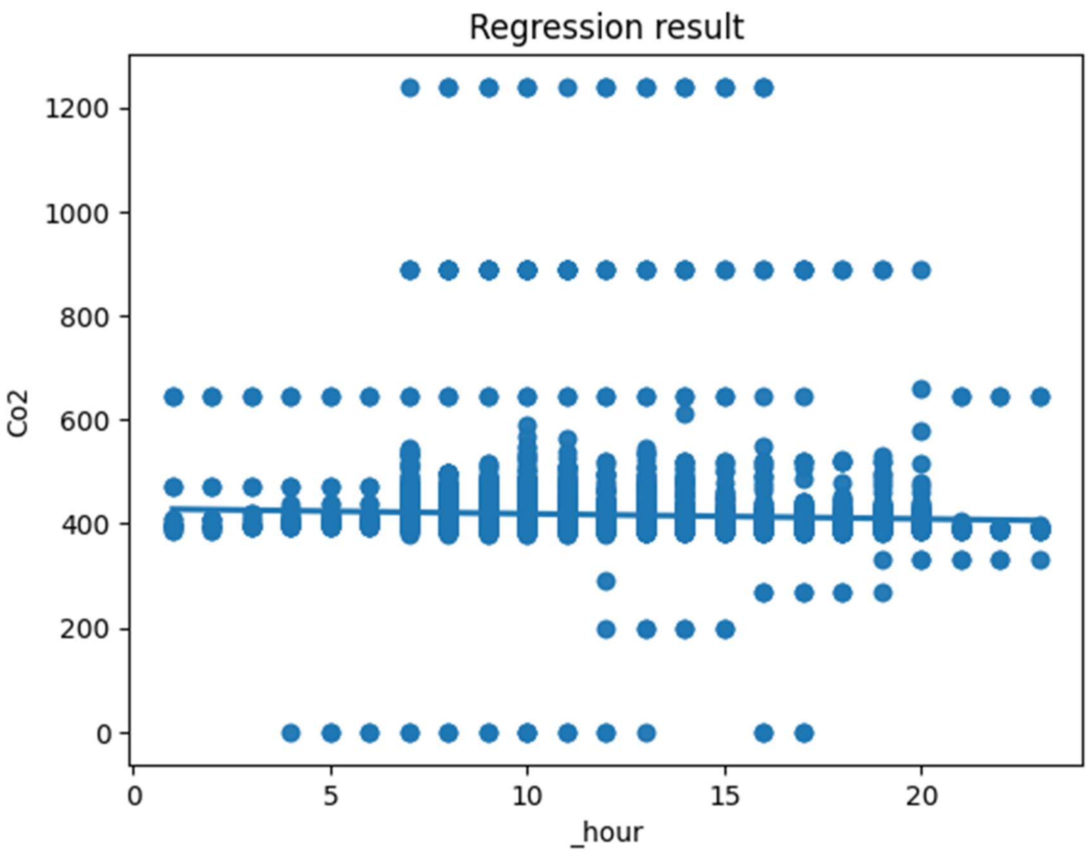


Figur 25: Winter season code: 11, Spring season code: 21, Summer season code: 31, Autumn season code: 41

This image 25 shows that the room air Co2 quality stays under 1000 PPM most of the time. In school buildings PPM under 1000 is required to be maintained all over the time. Like autumn Co2 level goes high compared to summer months.

Tested with plant mode Pre- comfort mode.

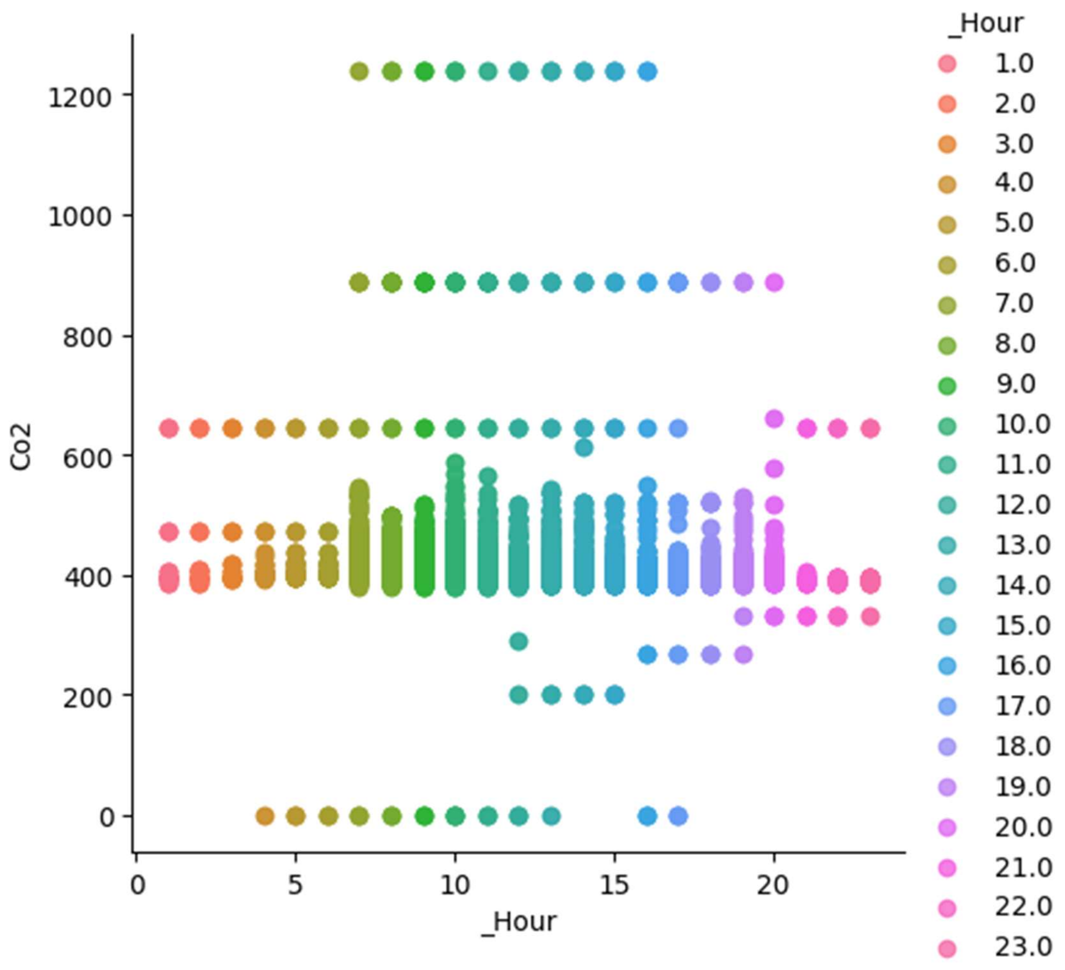
Before school hours generally plat mode is set to pre comfort and that maintains the Co2 levels under 600 most of the time. During night plant mode is **economy** and before school hours it moves to **pre comfort mode**. During school hours **comfort** mode is activated.



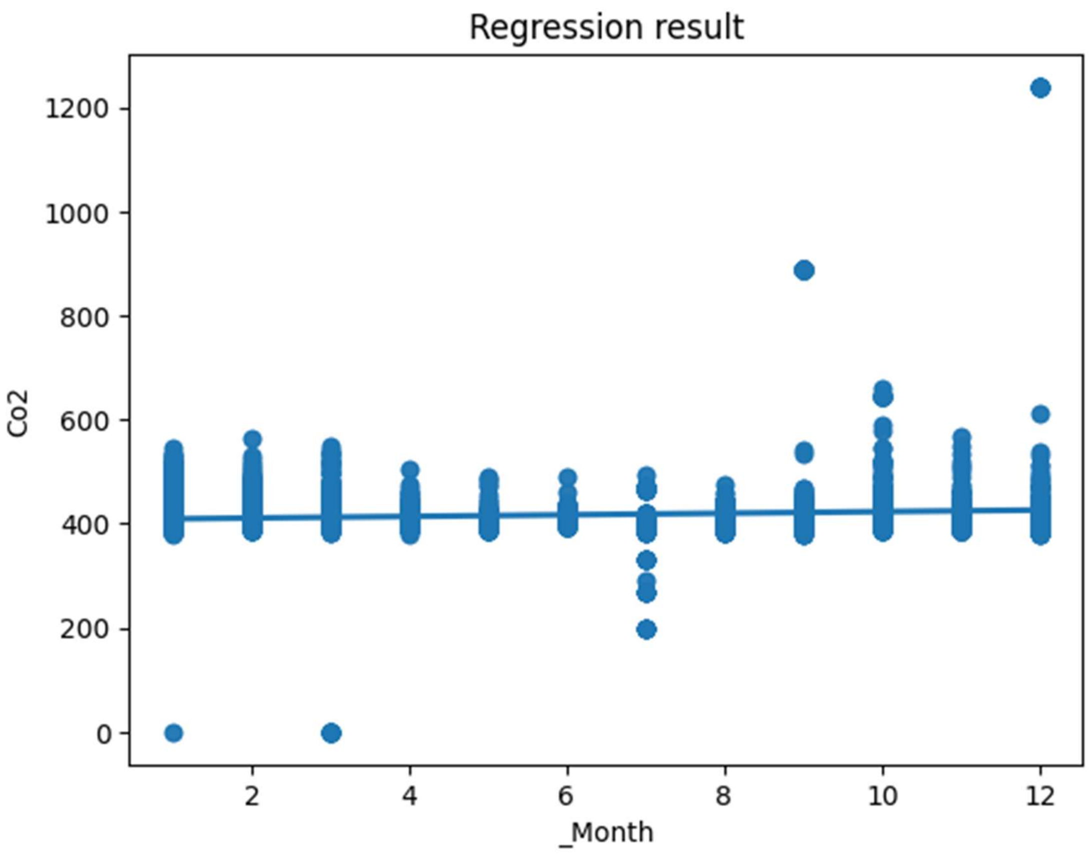
*Figur 26: When the plant operation mode in pre comfort then we can see that the Co2 is maintained within 400 to 600 PPM- **Pre- comfort mode***

Discussion:

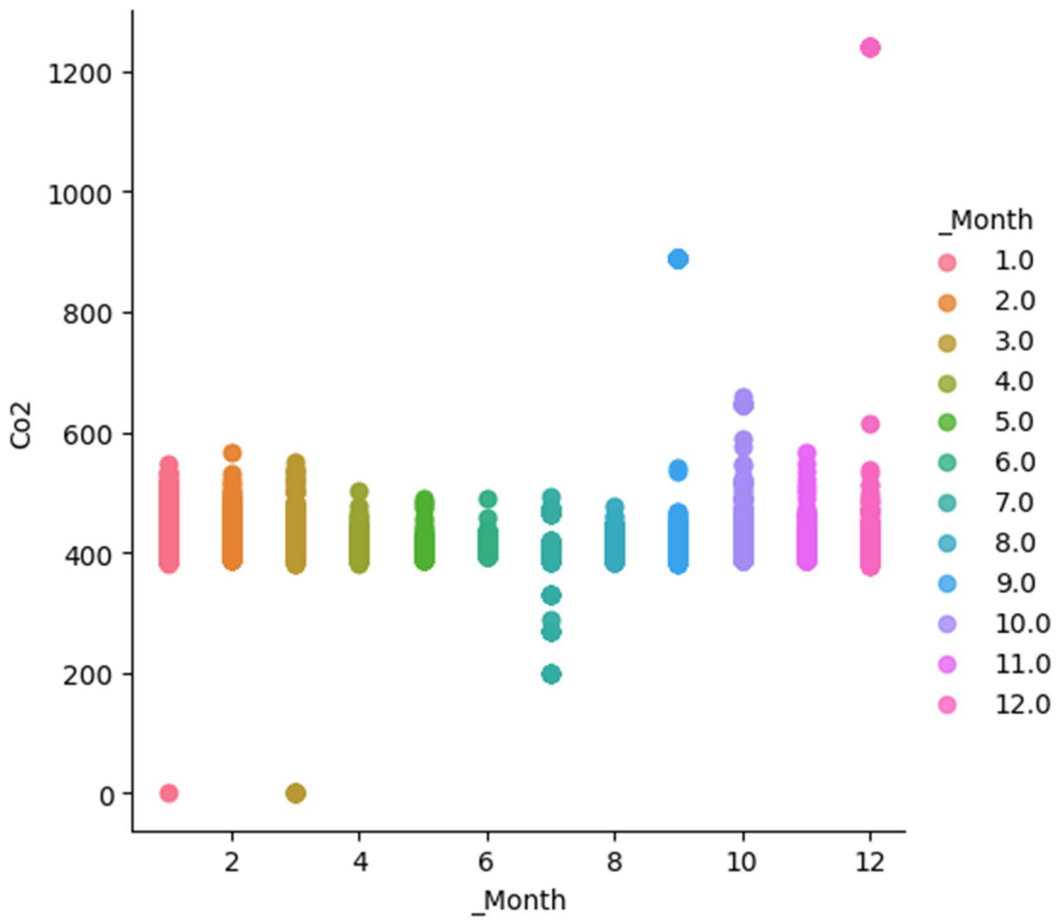
These plots show realistic situations based on historic data. It will be excellent to plot similar results form another room which one has more newer HVAC installation. Fine tuning and improvements are expected from a modern system compared to older system.



*Figur 27: Mostly it's stable Co2 level in classroom noticed- **Pre- comfort mode.** Dataset shows hourly report.*



*Figur 28: Mostly it's stable Co2 level in classroom noticed- **Pre- comfort mode.***



*Figur 29: mostly it's stable Co2 level in classroom noticed - **Pre- comfort mode***

Tested with plant economy mode.

Plant economy mode works outside working hours (between morning 16:00 to 08:00).

It is clearly visible that during daytime Co2 level is maintained under the control during the day but after office hours co2 level goes above 1000 PPM. (Image 30)

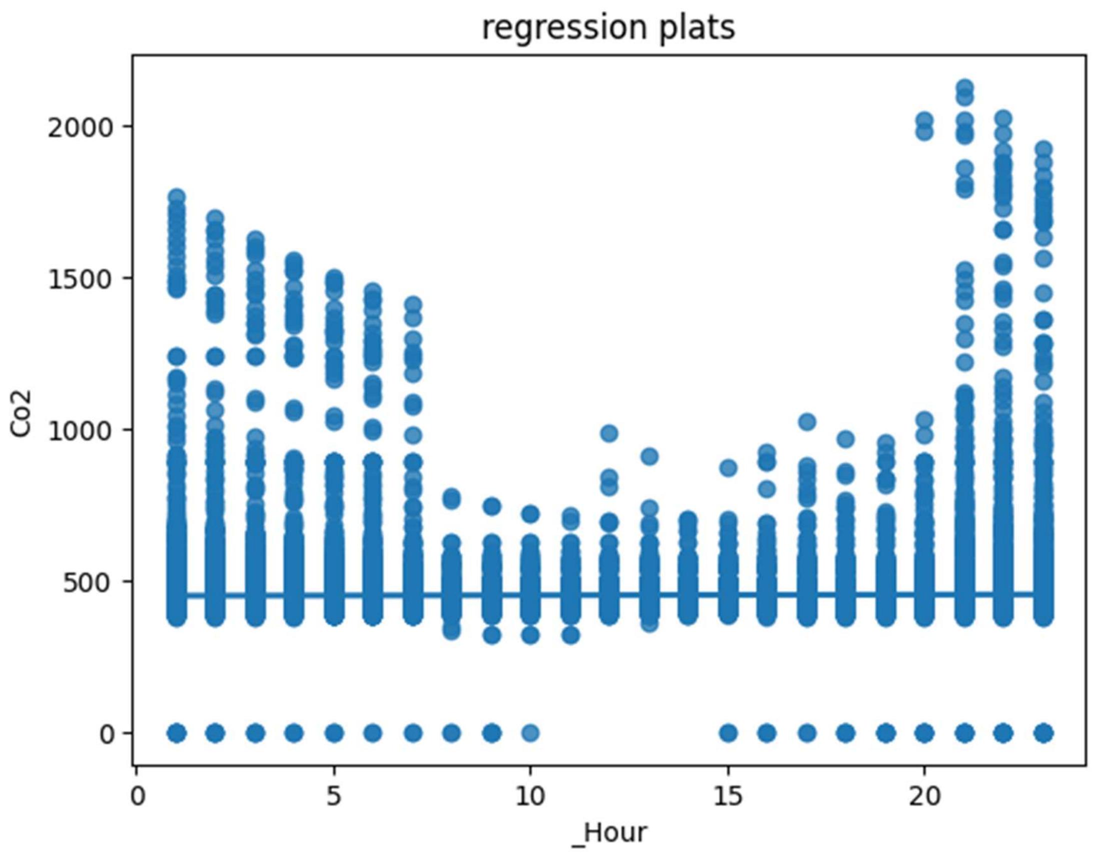


Figure 30: This is hourly Co2 levels in the room - economy mode. Between 07:00 to 15:00 hours it's very minimized Co2 level maintained.

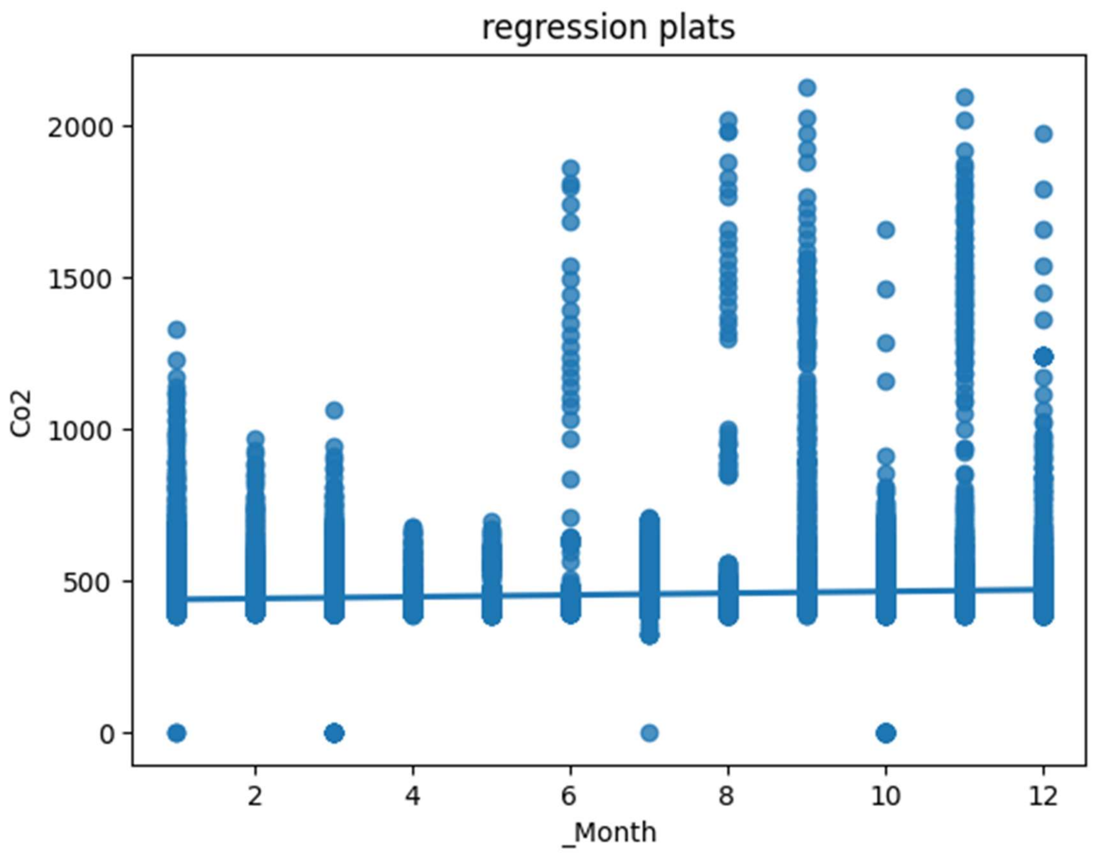


Figure 31: Month and Co2 - ***economy mode,*** *August onwards CO2 trends goes towards higher side.*

Tested with plant comfort mode.

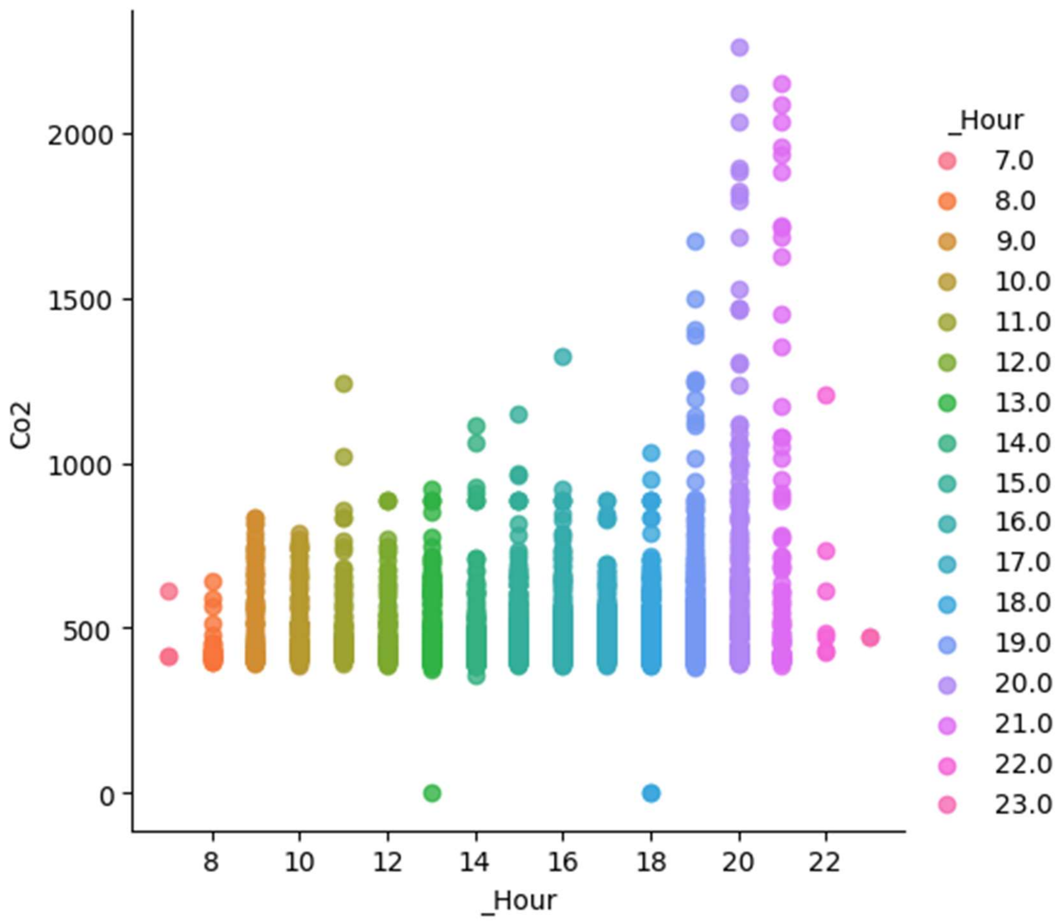


Figure 32: Plant comfort mode is active during school hours- ***comfort mode.*** *After 18:00 hours it is visible the Co2 level changes.*

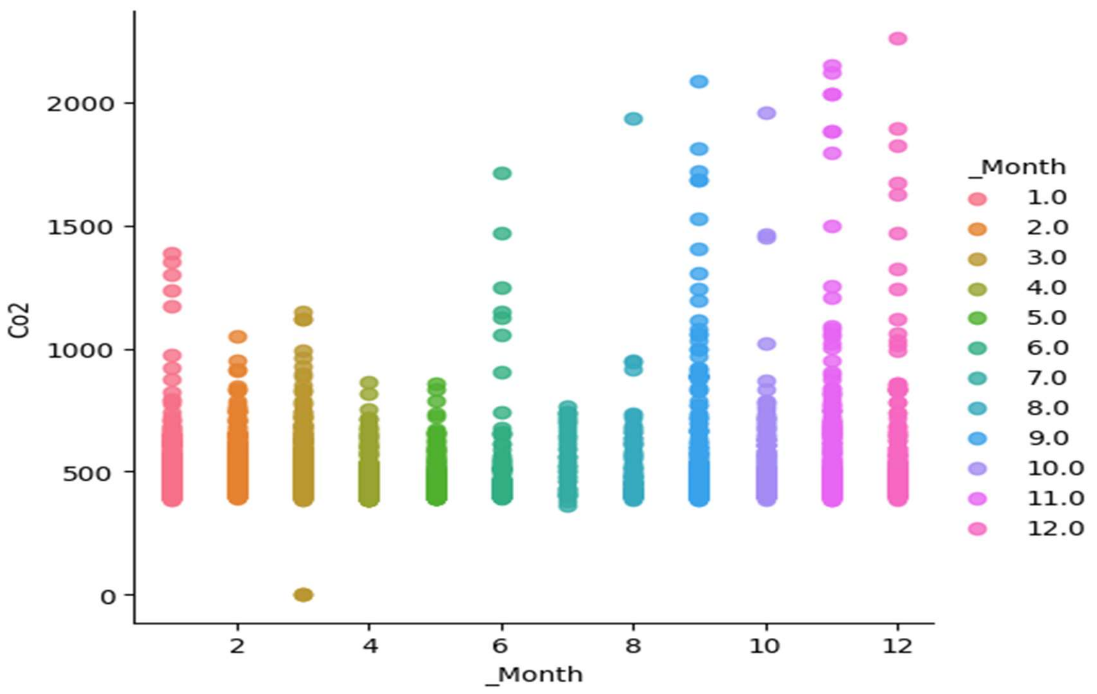
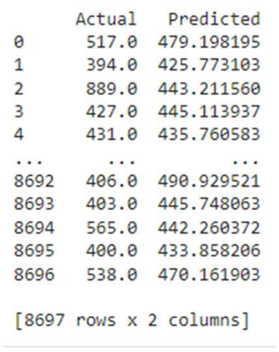


Figure 33: Plant comfort mode only active during day and then increase Co2 level noticed after September to January

Time series analysis with dataset

Linear regression with temperature and Co2 predicts Co2.

Linear regression is used to predict Co2 (PPM) values based on historical data.



```
      Actual   Predicted
0      517.0  479.198195
1      394.0  425.773103
2      889.0  443.211560
3      427.0  445.113937
4      431.0  435.760583
...      ...         ...
8692   406.0  490.929521
8693   403.0  445.748063
8694   565.0  442.260372
8695   400.0  433.858206
8696   538.0  470.161903

[8697 rows x 2 columns]
```

Figure 34: Actual and predicted value sample

Using the model Linear Regression, it identified the Actual and predicted values from the dataset.

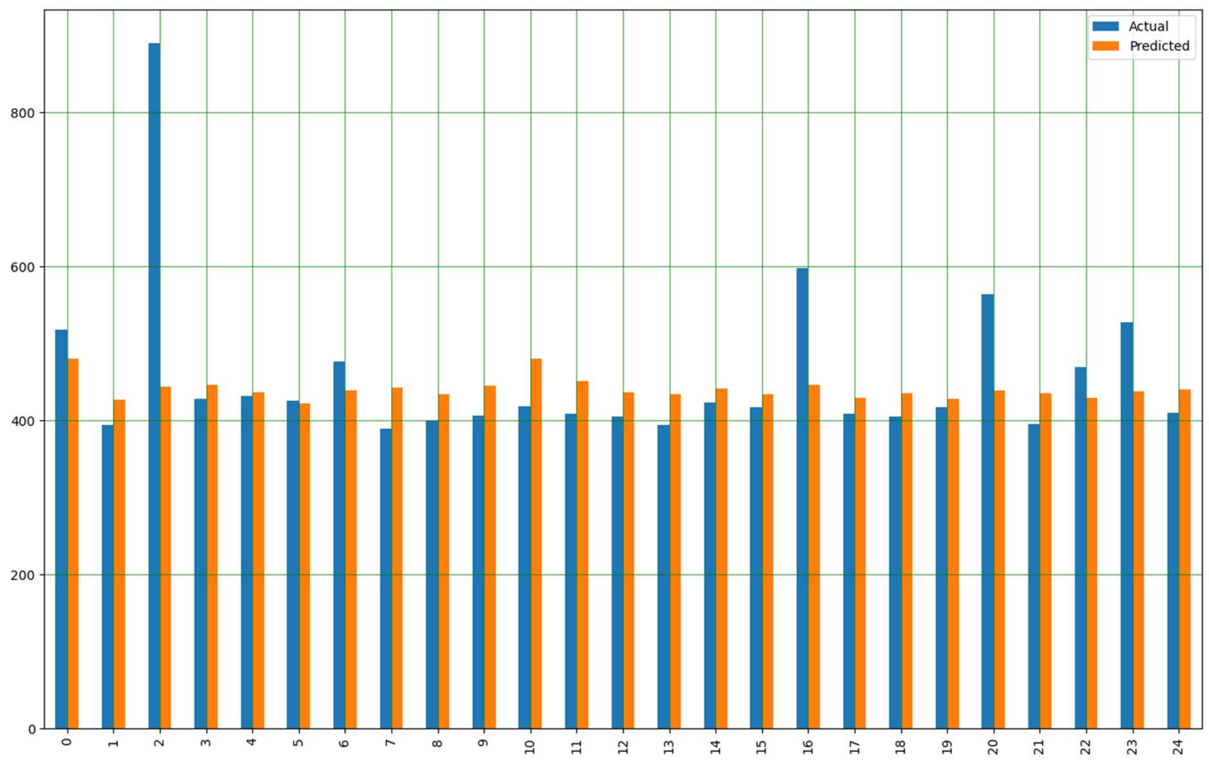


Figure 35: Plotting the actual and predicted values

Forecasting room temperature using time series.

Method

After data cleaning selected columns are picked to run the show from the dataset, learned from correlation map. Some of the fields like 'extn_temp', 'extn_pressure', 'extn_wind_speed', 'extn_wind_humidity', 'Supply_Air', 'Extract_Air', 'RoomTemperature', 'Co2', 'Radiatorventil' were picked to go further.

After nan value checking in the dataset, it was resampled with hour. As this prediction will be hourly. Again, NaN value checking done and substitute NaNs in a panda DataFrame with the values before or after it.

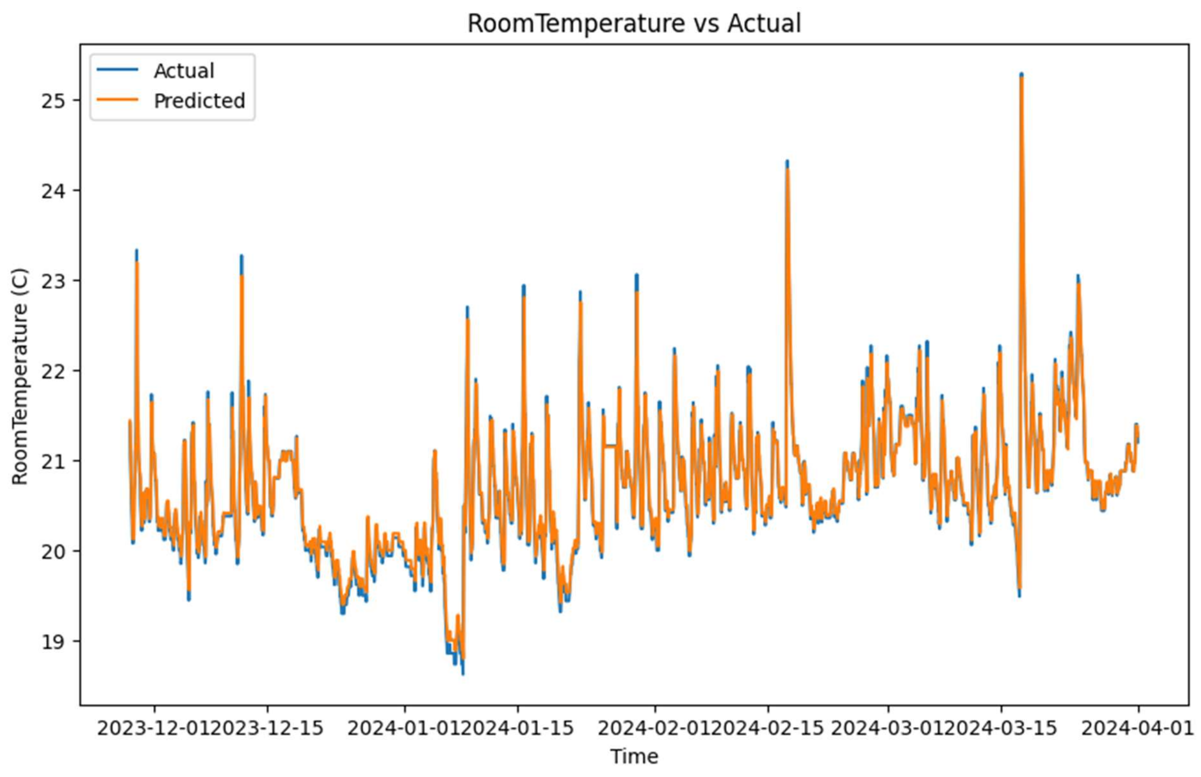
With MinMaxScaler data needs to be normalized again and sequence length and features are defined, and sequence and corresponding labels are created. Labels are converted to numpy arrays and spited into train and test sets.

Then sequestial model created with tensorflow and keras and identified tranable and non tranable params. Then with model?checkpoint best model created. Finally train the model with epochs=100. Then it’s time to evaluate best model on the test set. Plot training and validation loss values. [26]

From sklearn.metrics possible to get the mean absolute error (mea) and mean squared error (mse)and root mean squared error (rmse). There after actual and predicted values can be ploted with the selected columns.

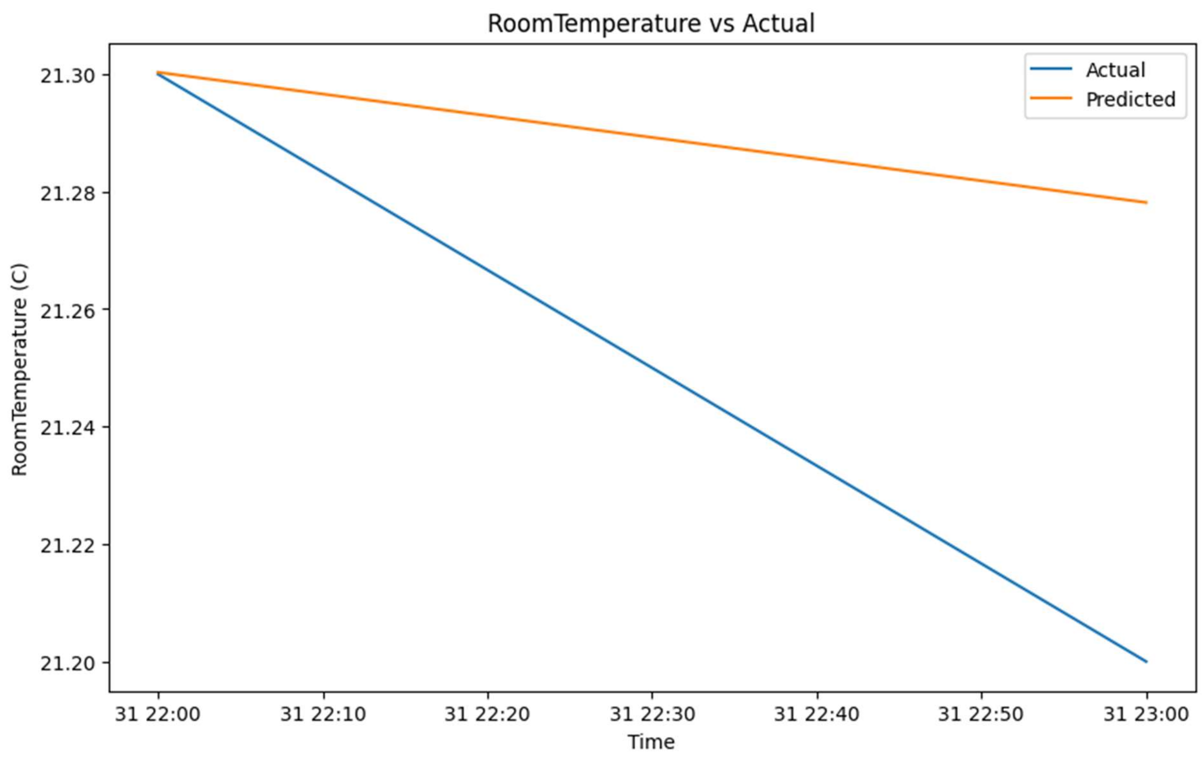
Result

This is the prediction of 125 days (3000 hours) and actual report.



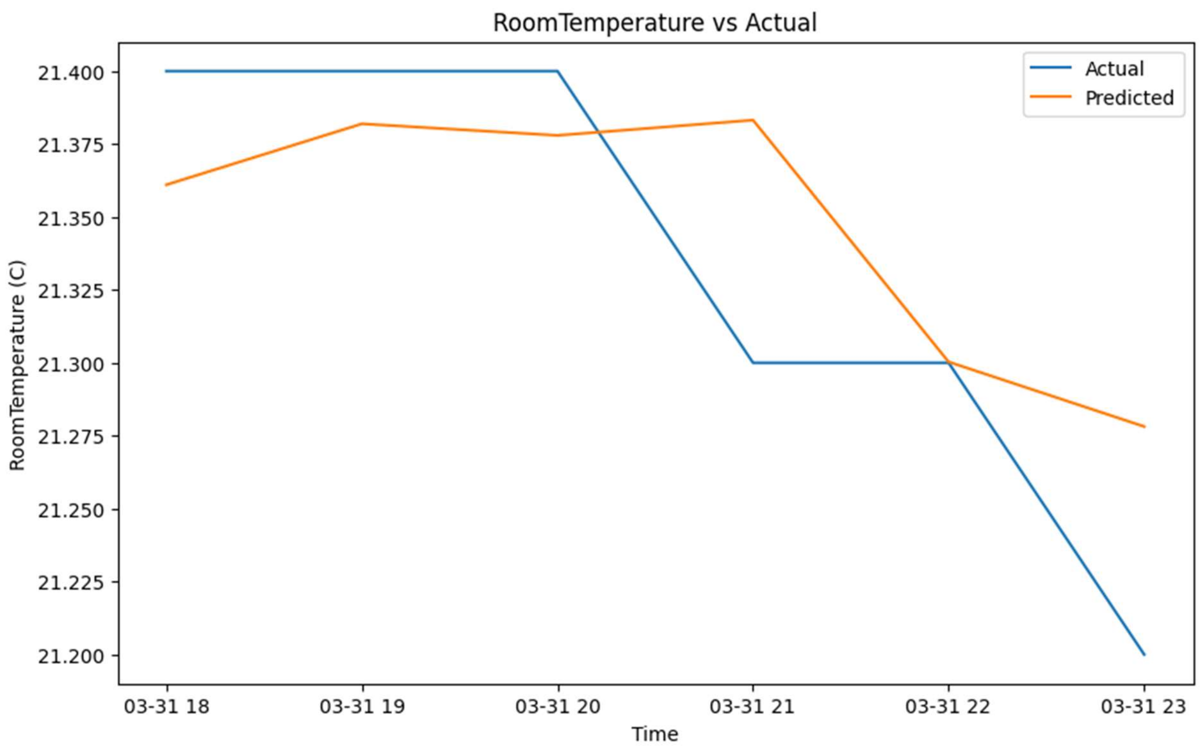
Figur 36: Predicted room temperature over 125 days

This is the prediction of 2 hours and actual report.



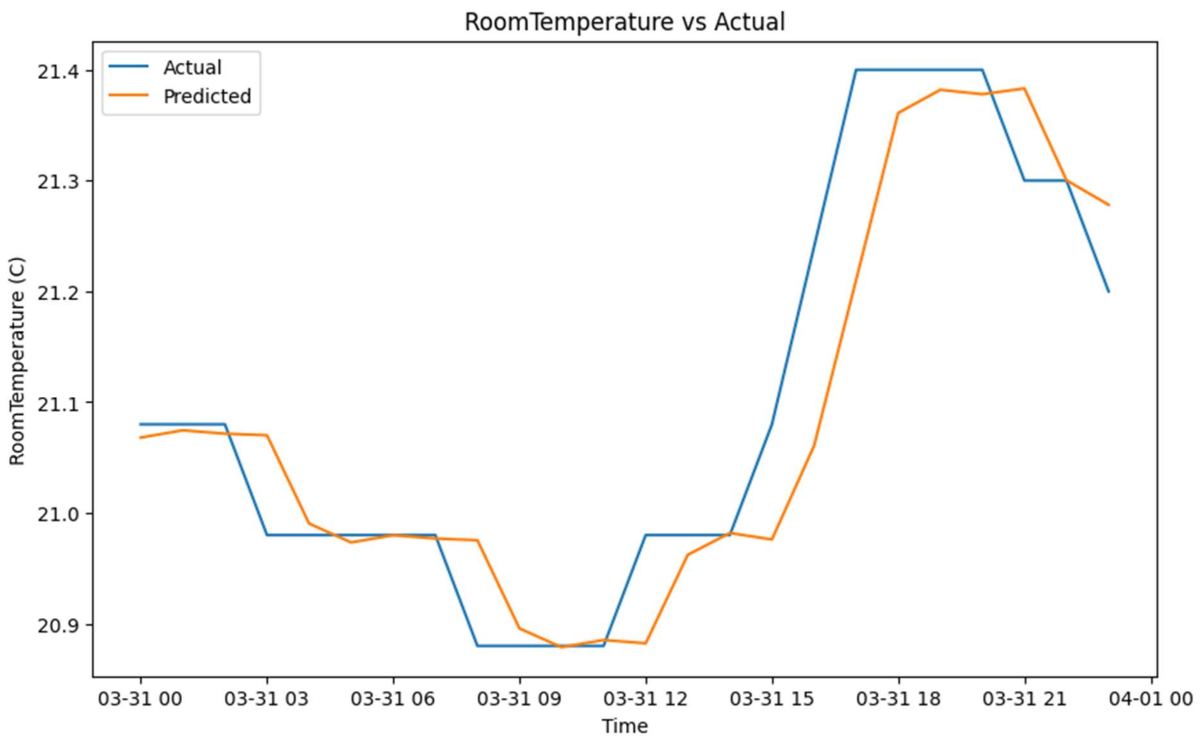
Figur 37: Predicted room temperature over 2 hours

This is the prediction of 6 hours and actual report.



Figur 38: Predicted room temperature over 6 Hours

This is the prediction of 24 hours and actual report.



Figur 39: Predicted room temperature over 12 Hours

Forecasting Co2 using time series.

This is the prediction of 24 hours and actual report.

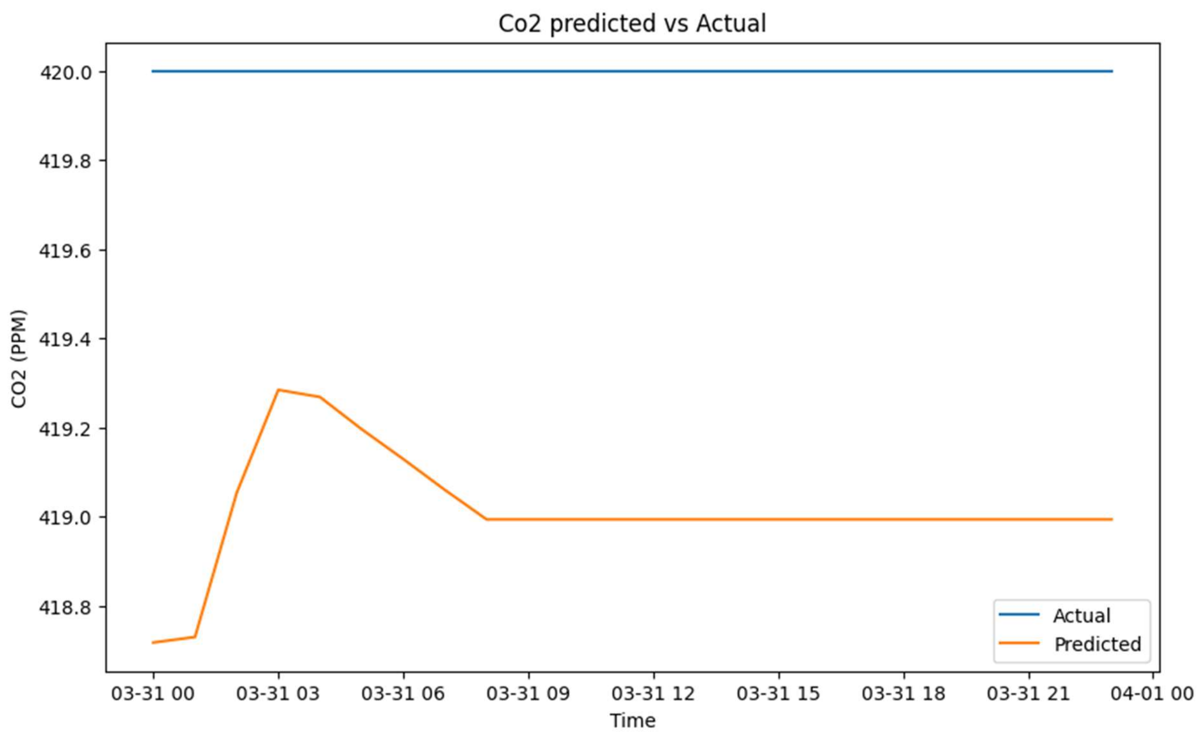


Figure 40: 24 hours Co2 level prediction

This is the prediction of 12 hours and actual report.

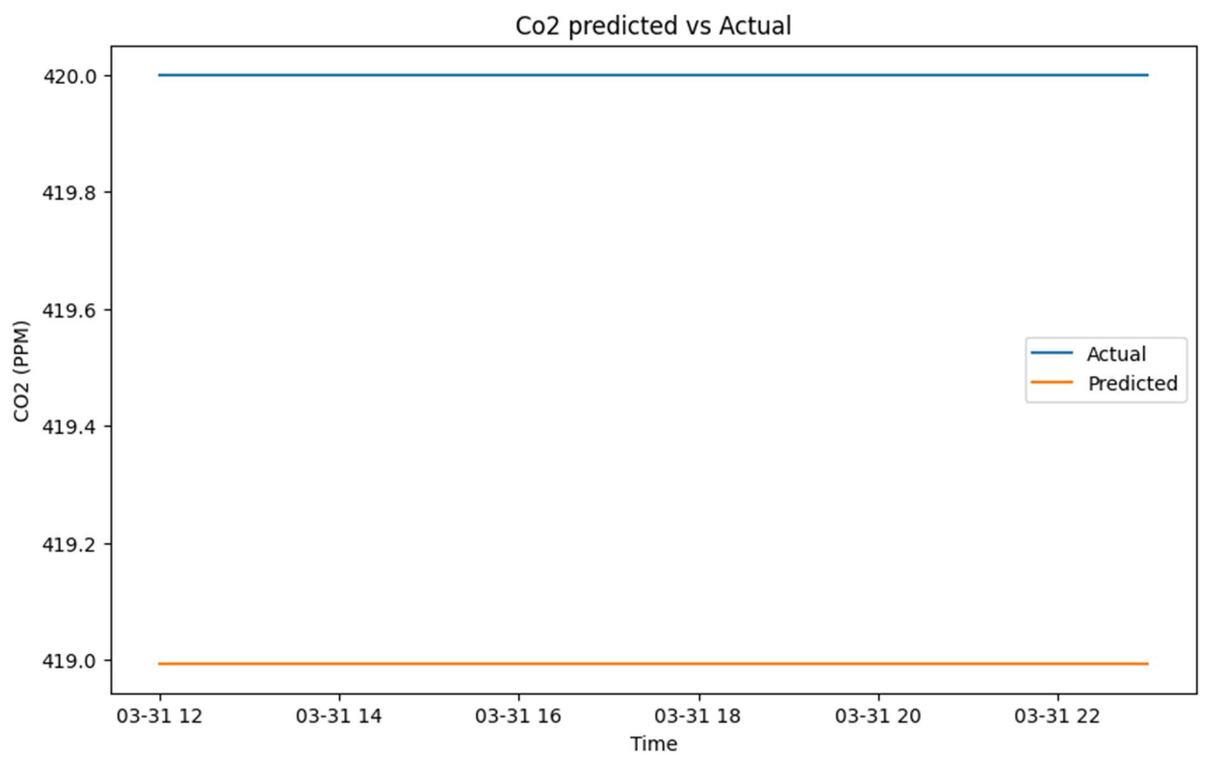


Figure 41: 12 hours Co2 level prediction

This is the prediction of 3000 hours (125 days) and actual report.

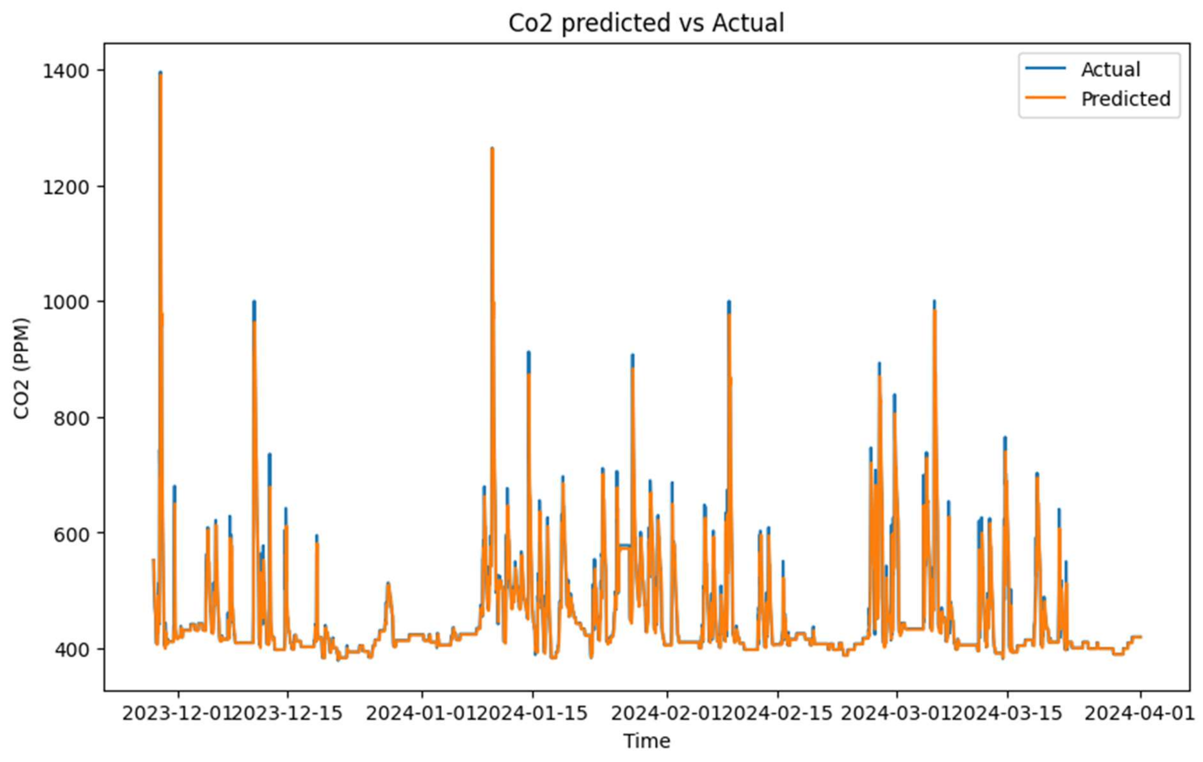


Figure 42: 125 Days Co2 level prediction

Forecasting Room temperature using multiple time series:

This uses the past data of multiple time series (e.g. predict indoor temperature using the past of outdoor temperature, humidity, heating on/off and so on.

This experiment done with some specific data columns like 'extn_temp', 'extn_pressure', 'extn_wind_speed', 'extn_wind_humidity', 'Supply_Air', 'Extract_Air', 'RoomTemperature', 'Co2'and ‘Radiatorventil'

Last 125 days actual and prediction

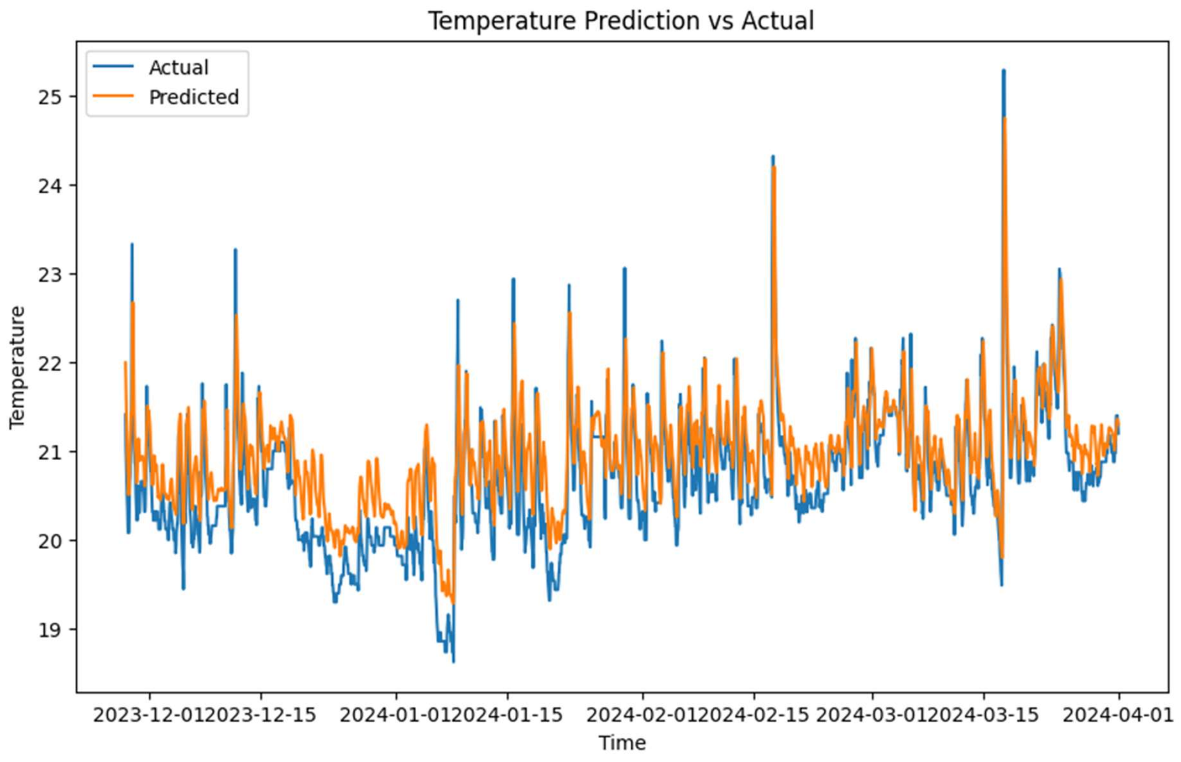


Figure 43: Temperature prediction with multiple time series columns 125 days.

This is the prediction of 2 hours and actual report.

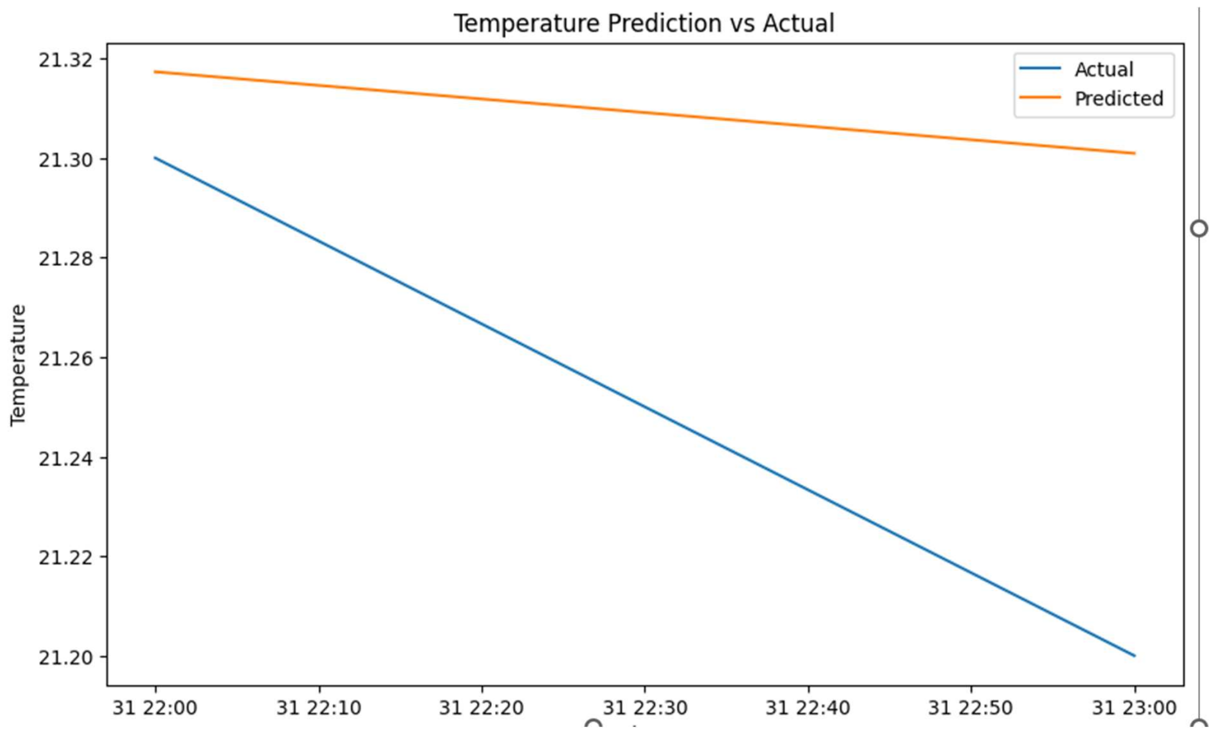
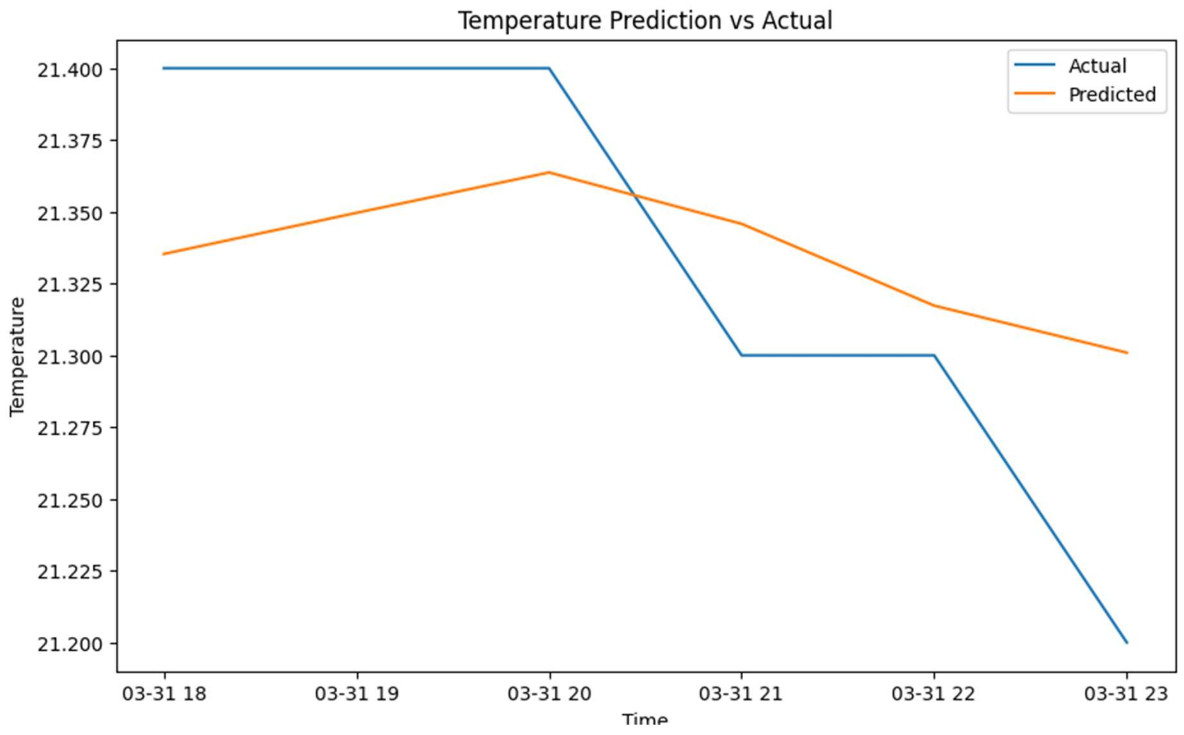


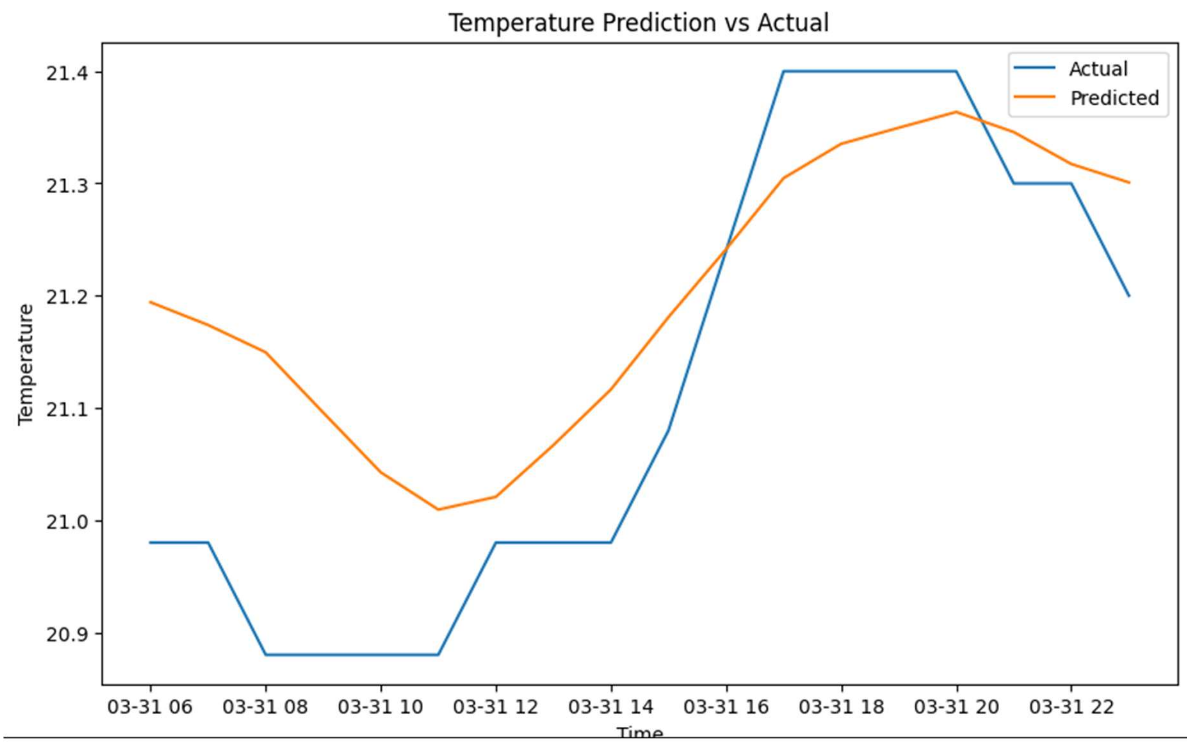
Figure 44: Temperature prediction with multiple time series for 2 hours.

Last 6 hours prediction



Figuer 45: Temperature prediction with multiple time series for 6 hours.

This is the prediction of 12 hours and actual report.



Figur 466: Temperature prediction with multiple time series for 12 hours.

This is the prediction of 100 hours and actual report.

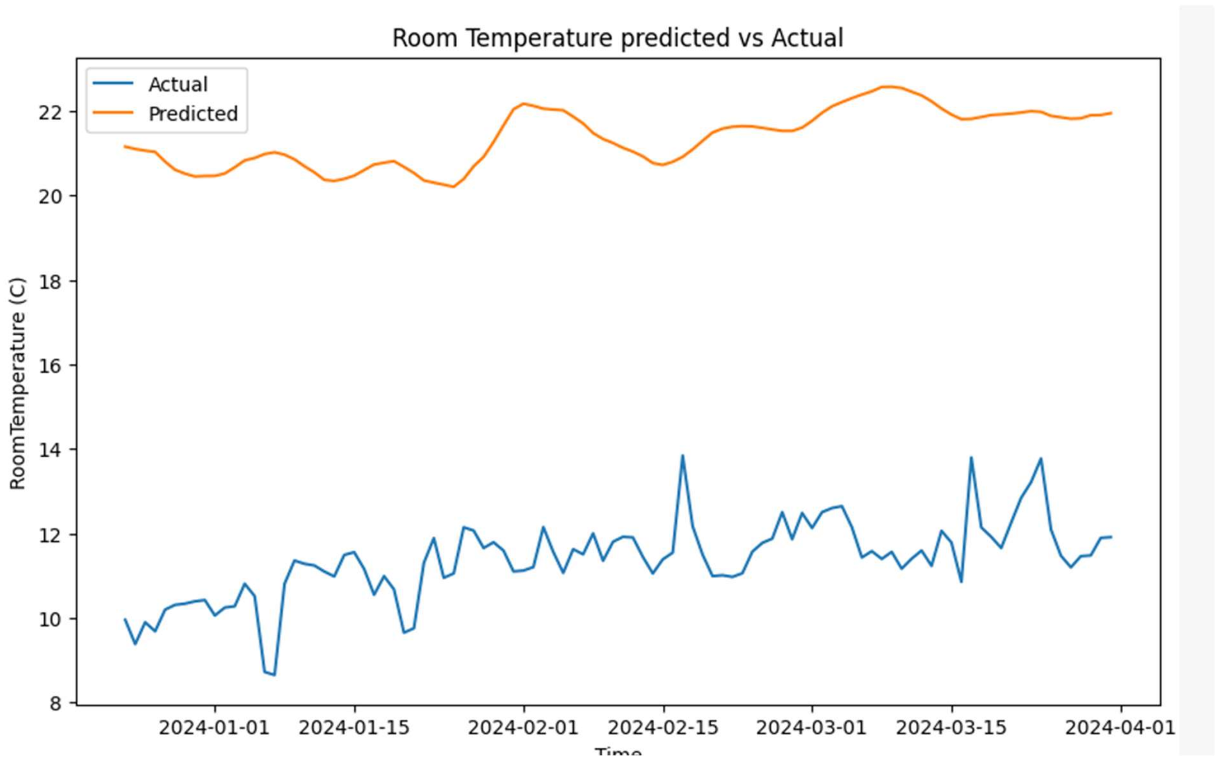


Figure 47: Temperature prediction with multiple time series for 100 hours.

Forecasting Co2 using multiple time series:

This is the prediction of 18 hours and actual report.

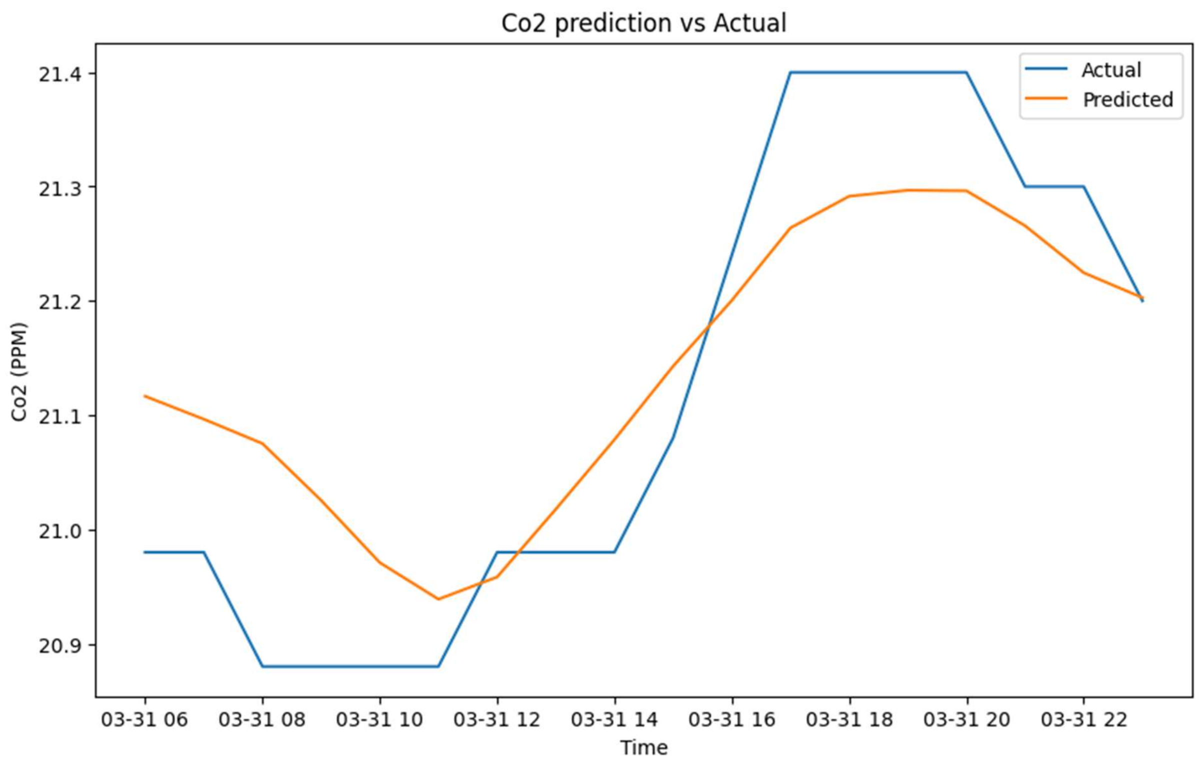


Figure 48: Co2 prediction with multiple time series for 18 hours.

This is the prediction of 12 hours and actual report.

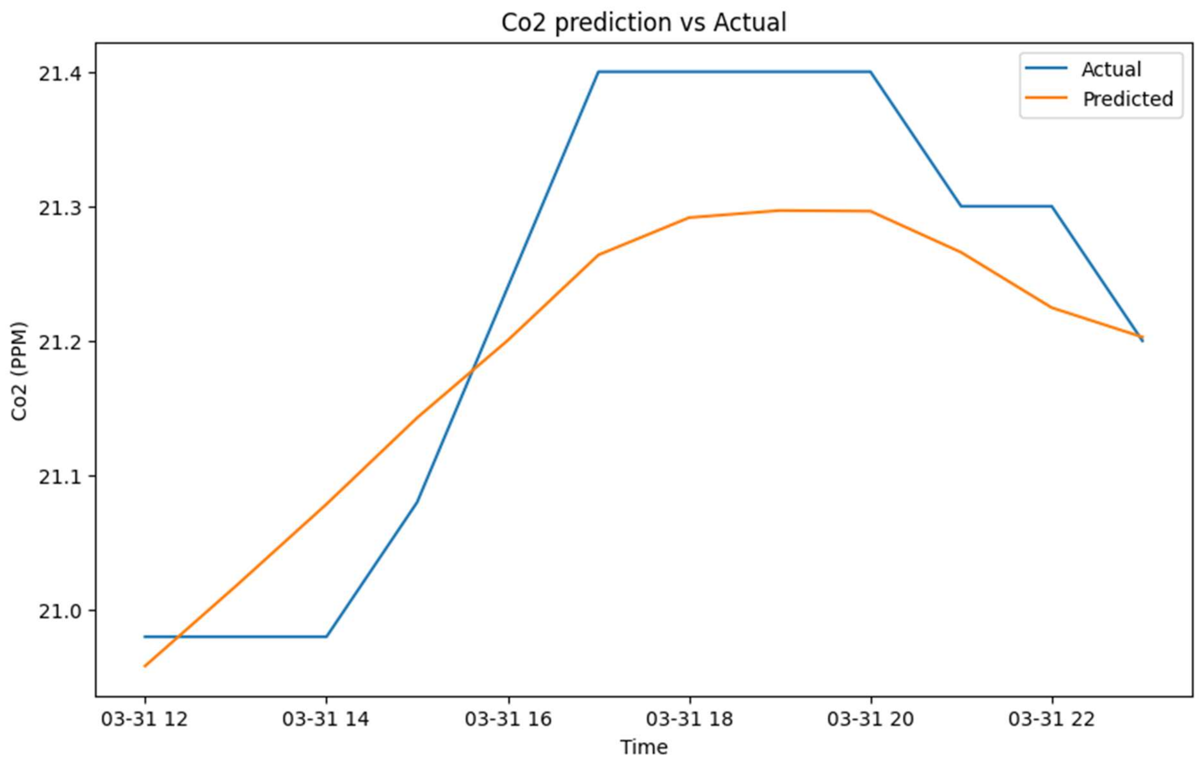


Figure 49: Temperature prediction with multiple time series for 12 hours.

This is the prediction of 2 hours and actual report.

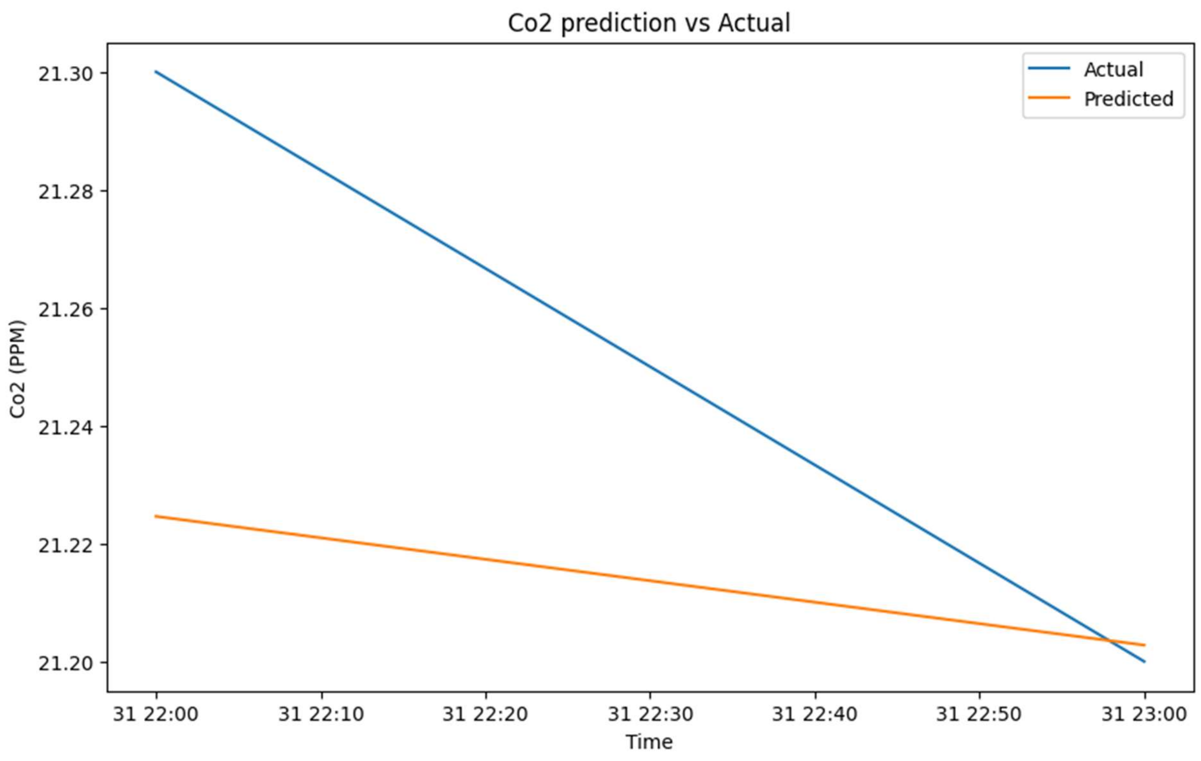


Figure 50: Temperature prediction with multiple time series for 2 hours.

This is the prediction of 125 days and actual report.

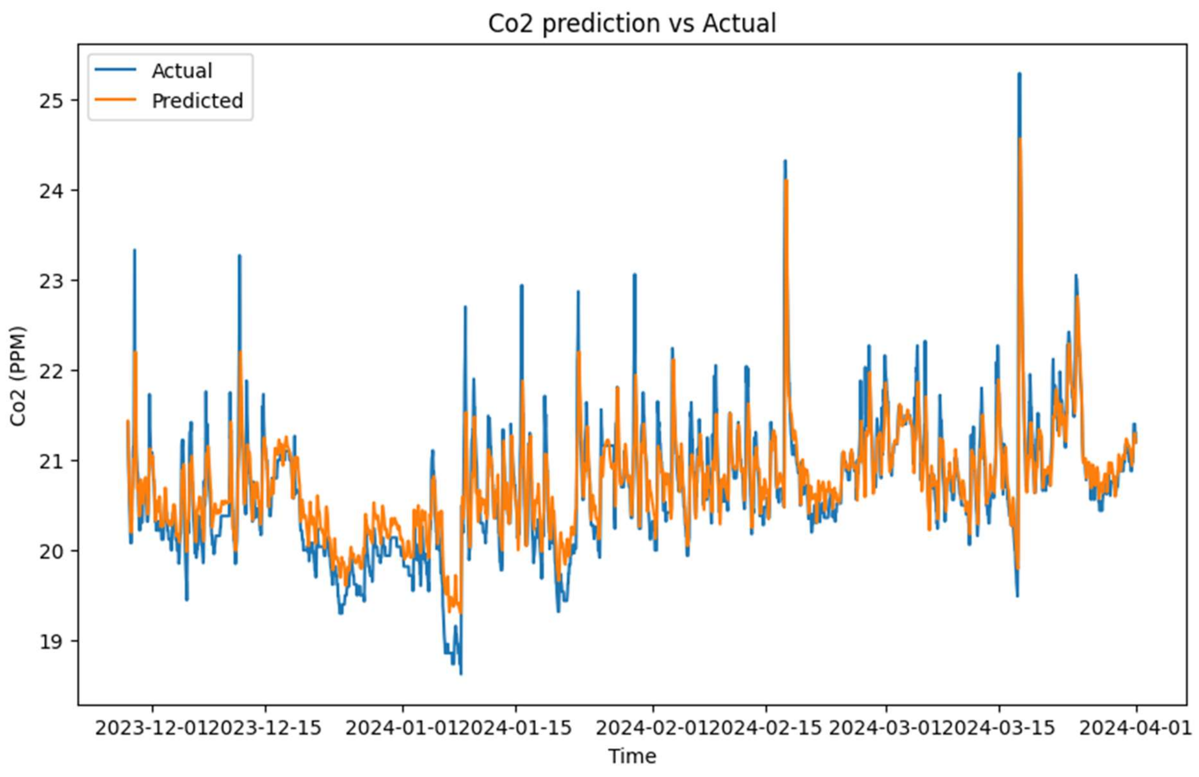


Figure 51: Temperature prediction with multiple time series for 125 days.

Forecasting using multiple time series (TensorFlow)

Multivariate forecasting is a statistical technique that uses future values for multiple connected variables at once. Multivariate forecasting begins with collecting historical data for each variable to identify trends, establish correlations based on relationships, and project future values. [27]

Common techniques used in multivariate forecasting are **structural equation modeling** (SEM), which allows the investigation of complex interactions between variables, and **vector autoregression** (VAR), which evaluates the interdependencies between several time series variables. Furthermore, machine learning techniques like neural networks and gradient boosting machines also have become increasingly popular in multivariate forecasting tasks due to their ability to identify complicated patterns and nonlinear correlations in data [28][29]

Multivariate is predicting several variables over time provides a comprehensive understanding of data dynamics. Consider a Weather forecast that includes lowest temperature, the daily highest temperature, humidity, and more, in addition to wind pressure. This degree of specificity is added to our data forecasts using multivariate forecasting.

Long Short-Term Memory (**LSTM**) have memory powers that enable them to recognize complicated correlations in our data, unlike ordinary algorithms, they are ideal for deciphering complex multivariate patterns.

TensorFlow platform's high-level API is called **Keras**. With an emphasis on contemporary deep learning, it offers an easy-to-use, extremely productive interface for resolving machine learning (ML) issues. It is possible to perform data processing, hyperparameter optimization, and deployment. [30]

Methods

Initially dataset was loaded. It is important to clean the row dataset and remove all space and null values. There are 24 columns in the data set but actual **air supply and Co2 prediction** and **Plant operation mode and room temperature** in time series is the initial goal. Data frame needs to be filtered with those columns. So, preparation and preprocessing of data is crucial.

Frequency conversion and time **series resampling** is important. In place of missing numbers, use a constant value or a descriptive statistic (such as the mean, median, or most common) along each column. To have more precise forecasts and quicker convergence feature scaling is required.

Dataset **normalization** is the process of transforming features to a common scale. This enhances the model's functionality and training stability. **SingleStepSampler** is used to make the dataset ready to prepare the dataset for single-step time-series.

SingleStepSampler requires two inputs: a window size and a dataframe, represented as df. Two lists, called xRes and yRes, are initialized in the function to hold goal values and input characteristics, respectively. Based on the specified window size, the function iterates across the rows of the dataframe using two nested loops to create sequences of input features (xRes) and corresponding target values (yRes).

Divide the dataset in an 85:15 ratio between the **training** and **testing** sets. This is data Splitting.

Using **TensorFlow's Keras** API, a multivariate Long Short-Term Memory neural network model is created. The model represents a linear stack of layers and is first initialized as a sequential model. The units of the LSTM layer in the architecture are intended to process input sequences whose shapes are determined by the quantity of features (columns) included in the training data. A dropout layer is added to stop overfitting.

This output layer's activation function is set to linear. The model is assembled using mean squared error as the loss function during training and mean absolute error (**MAE**) is used as a metric for additional assessment. The training procedure is aided by the utilization of the **Adam** optimizer.

This code segment employs an LSTM model to visualize the multivariate time-series **forecasting** results. The 'Date' column is used as the index when the dataset is first loaded again. The index is adjusted once the **date** column is transformed into a datetime format. The test set values are predicted using the LSTM model. A containing the **predictions** and the **actual** values is created. This Data Frame is aligned with the original **dataset** by being allocated the correct date index. Next, a Matplotlib plot is made to display the expected and actual values over time.

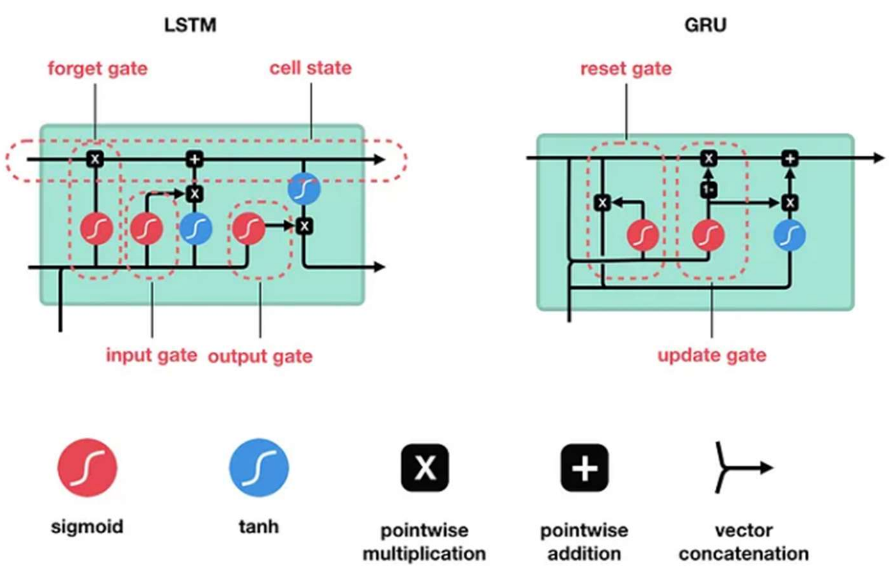


Figure 52: Short-term memory was addressed with the creation of GRUs and LSTMs. They possess internal components known as gates that can control the information flow. [31]

Advantage and disadvantages:

Works with multi-layer networks, they may capture intricate temporal dependency patterns.

Long-term dependencies can be effectively captured by LSTMs, which can also handle extremely nonlinear and non-stationary data.

They can also pick up on time dependence and trend terms in time series data.

LSTMs can be coupled with other models, such CNNs, to capture local trend aspects as well as long temporal dependencies.

It's complexity, noise, and disorderliness of data can make prediction difficult. LSTM architectures and fine-tuning parameters can be difficult and need human supervision.

Outcome

Predict HVAC Air supply and predicted Co2:

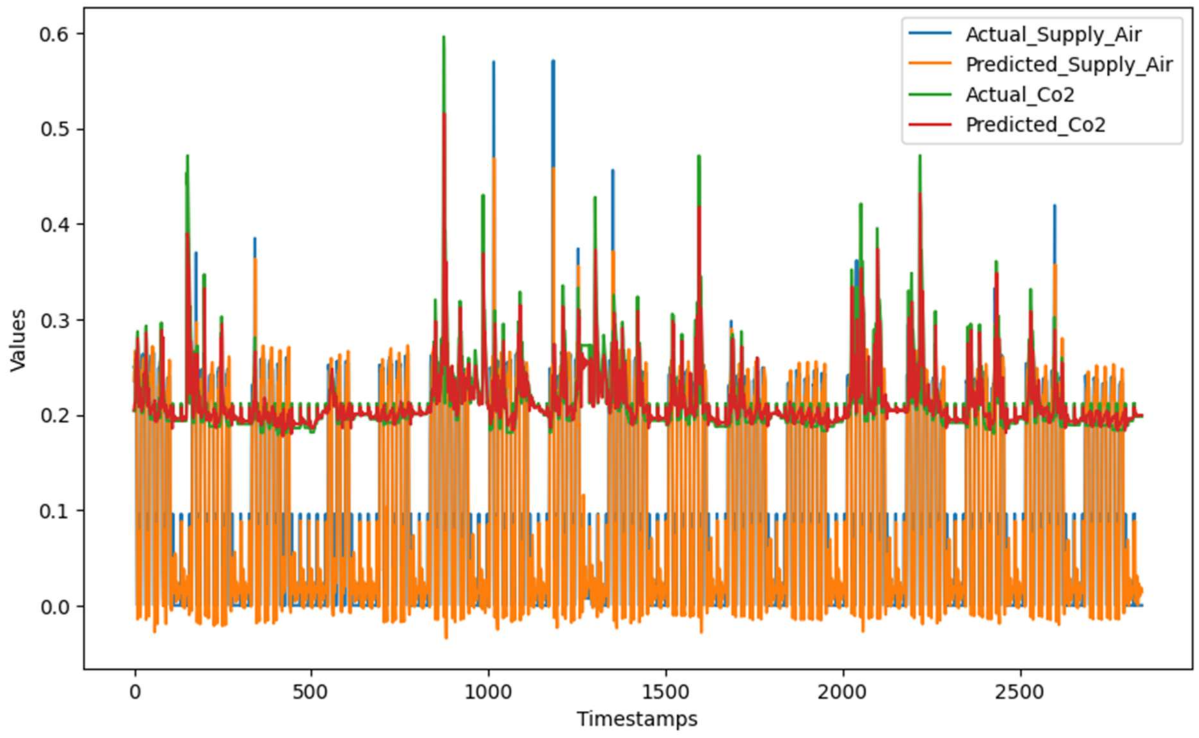


Figure 53: Air supply and Co2 predicted with another co- related variables.

Predict HVAC plant mode and temperature:

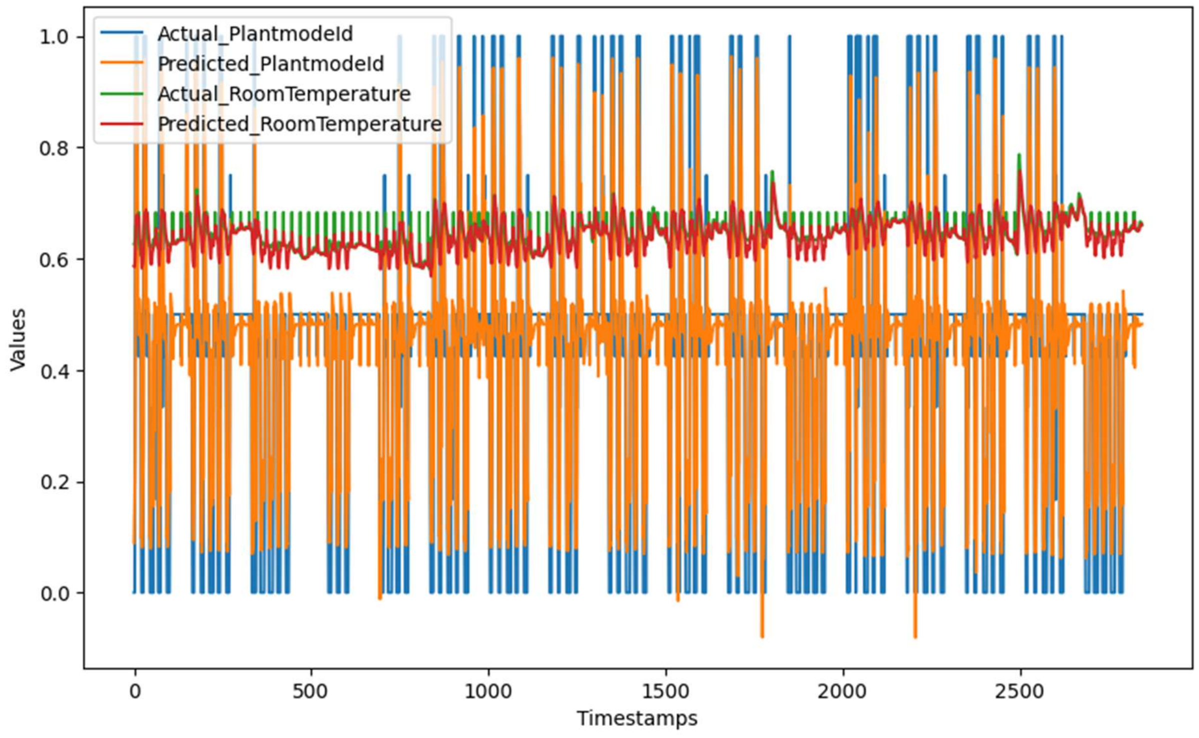
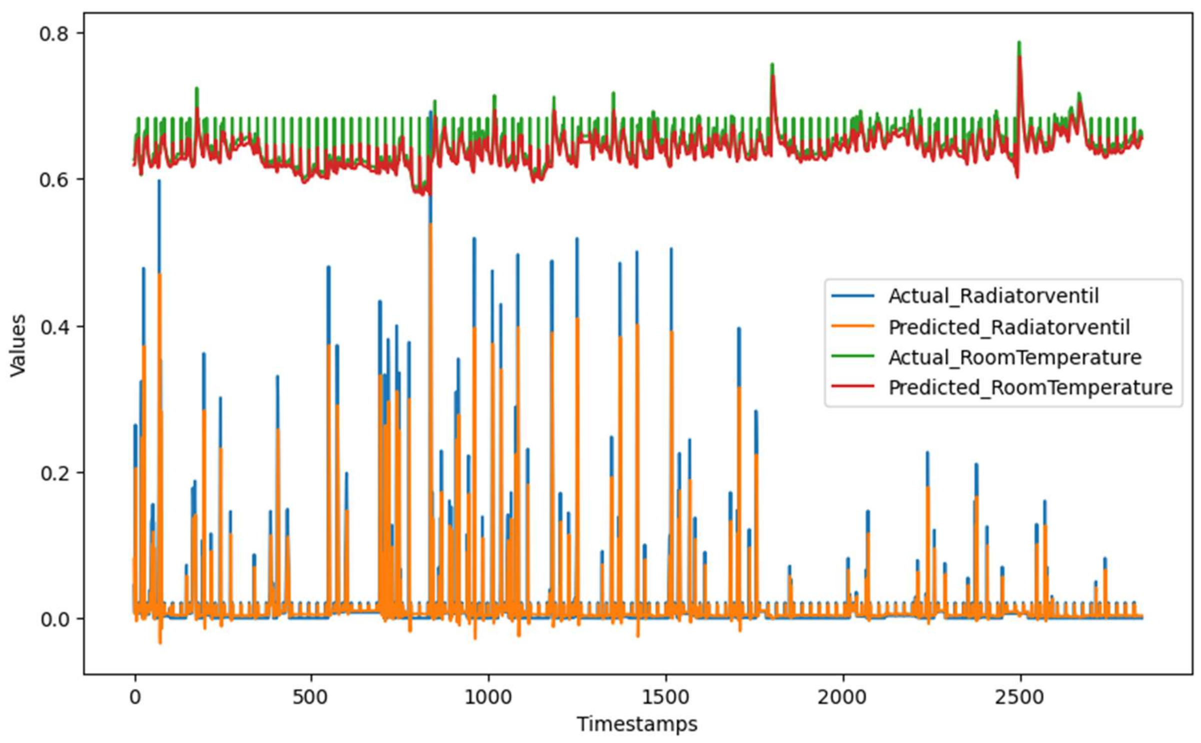


Figure 54: Plant operating mode and Co2 predicted with another co- related variables.

Predict HVAC plant radiator control and temperature:



Figur 55: Radiator and room temperature predicted with another co- related variables.

Model Evaluation

Possible to monitor predictor variables. When we see the error is very low then we can conclude that the LSTM model is performing very efficiently. It is possible to make more effective with advance loss reduction and hyper parameter tuning.

Summary and conclusions

Idea of this thesis writing started from some practical aspects. Oslobygg has real time challenges like controlling the overall situation and checking the system report in an organized way.

Massive building infrastructure, sensor configuration files, real time IOT data monitoring and data prediction are few areas where Oslobygg wants to investigate. It's important to check if all the devices are performing as expected. Like if the temperature and humidity is perfect and stable in a classroom over the period and to check if end of the day HVAC can change it's status to economy mode to save energy.

Today there are some advanced sensors that can control heating and ventilation systems but those are only implemented in some limited buildings. The renovation job is expensive with respect to money and time. All old HVAC set ups are mostly manual and runs on fixed settings. Some runs with fail configuration like heating and cooling runs parallel.

Major challenges were to collect data and understand the aspects and relations between them. Multiple vendors, too many parallel technology and tools are implemented over the time. These IOT data are stored in different databases. Some uses SQL server[36], some uses Postgre[37] or something else. Identifying correct people to take out some sample data was important. Security issues and permission were involved and some consulting cost for few hours was also involved. After several meetings row data retrieved from vendor database. Many discussions were required to know how the setup is working.

Data collection a storage and transformation got high attention.

It was important to thing about the concept of Datawarehouse, where all transactional data will arrive eventually with historical data. Data transformation and data merging done conceptually.

Data cleaning, feature engineering, null checking, understanding the correlation between different component and checking the heatmaps.

Anomaly

Testing with different regression plots and checking anomaly is very crucial when it connects with school kids. Several algorithms tested and performance evaluation done. It connects with realistic day to day situations in school. Dataset contains real data and expectation is to get real time scenario-based regression plots and tested with different situations and circumstances. Realistic result obtained.

Time series forecasting.

Worked with time series forecasting for temperature and Co2 level. That shows hourly prediction from historical trained dataset. Tested Multivariate time series prediction with different HVAC component level.

This thesis is the base on real time challenge and real data set used to figure out possible relations between different components and predictions, failures. This is focused to build a new application that solve most of these problems and improve the system performance with better control.

Contribution and further work

Oslobygg pays millions for consulting, implementation, running cost and electricity bills for running HVAC systems. Mostly these are fully controlled by vendors and few responsible people of Oslobygg, they never receive overall consolidated report or analysis data for some action. It's fully manual reporting system and mostly errors and issues are encountered during maintenance checks or when something is probably completely down. Failure alerts from HVAC system are also chaotic and impossible to know whether that is false or real without human interaction.

Main reason of this is insufficient control on data and usage. Expectation is to have an upcoming tool where we go through data and predict situation and automated / manual action making decision will be faster.

Example:

Some buildings have old HVAC set up. Is HVAC running actively after 18:00 on a Friday evening? AI can find that and ask to stop or change the settings automatically to minimize energy use.

If the room temperature is already 25+ then heating radiator can receive signal to reduce the heat supply instead to activating the cooling agent.

There are endless situations and challenges that can come up when base prototype and one MVP (Minimal viable product) will be demonstrated with-in Oslo kommune, because HVAC experts, building engineers, maintenance engineers, automation engineers will come and discuss practical challenges and what can be immediate and long-term solutions for that.

Proposal

Oslo kommune uses Microsoft as a platform and technology [34]. Modern tools like Microsoft fabric [33] and Microsoft AI will be used to build the prototype in coming future.

Considering a smaller number of AI skilled resources in Norway, it is difficult to build and maintain the open-source solutions. Specially the maintenance part is extremely important. The goal is to deal upcoming AI based cases to reduce energy usage, error handling and better maintenance with experience gained from this area.

This thesis will surely show the roadmap to use huge source of data and find out different possibilities. Internal discussion will boost confidence among other area experts and that will bring more questions and discussions.

IKT Oslobygg arrange presentation and discussion sessions with other building technology area experts and use of AI is very positive initiative. Demonstration with HVAC system data will be extremely interesting topic for them.

Technology Roadmap

Oslo commune and Oslobygg has already adopted some of the tools from this sample architecture. In some areas like Finance, HR, and project management they have integrations with different applications and use them for further report and customization. Very often we use AI or best to say Oslo Kommune started using AI slowly.

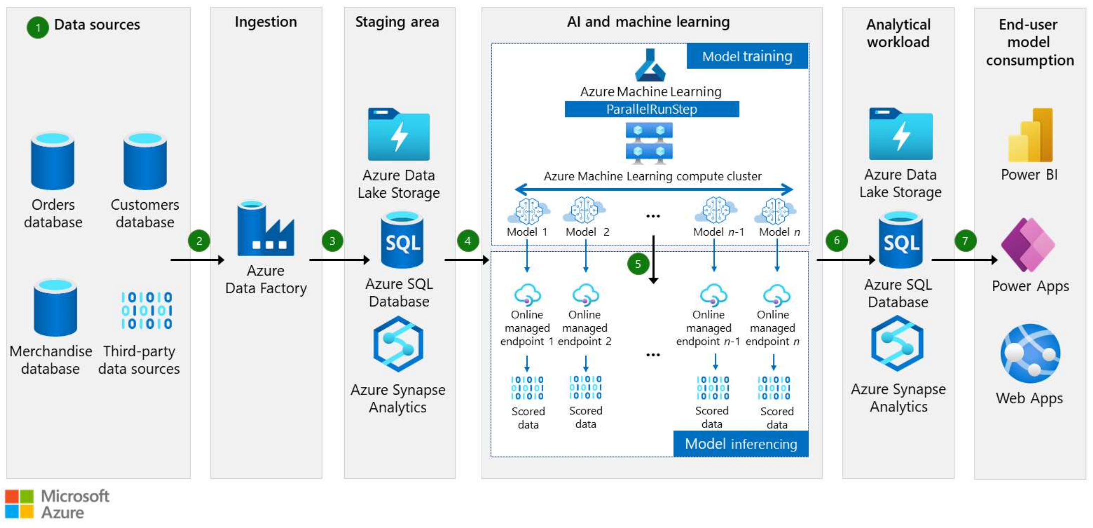


Figure 56: Azure Architecture [35]

HVAC data is completely missing there yet because it is complex to connect and gather them, it is unexplored but core part of our building infrastructure. It comes in large volume sensor data. Long way to go and hope this initiative will boost some energy to go further. Maybe Oslobygg end up using Azure IOT hub [38] and Microsoft AI soon.

Reference

[1]	https://www.oslo.kommune.no/etater-foretak-og-ombud/oslobygg-kf/#gref	2024
[2]	https://www.forbes.com/home-improvement/hvac/how-do-hvac-systems-work/	6, Nov 2023
[3]	The method of evaluating operation performance of HVAC system based on exergy analysis; Bo Fan, Xinqiao Jin, Xing Fang, Zhimin Du https://www.sciencedirect.com/science/article/abs/pii/S0378778814002795	Vol 77, July 2014, Pages 332-342
[4]	ASHRAE CO2 Standards for Classrooms https://www.co2meter.com/blogs/news/ashrae-co2-standards-classrooms	August 21, 2023
[5]	Optimising building heat load prediction using advanced control strategies and Artificial Intelligence for HVAC system. Osama Khan [a], Mohd Parvez [b], Mohammad Seraj [b], Zeinebou Yahya [c], Yuvarajan Devarajan [d], Beemkumar Nagappan [e] https://www.sciencedirect.com/science/article/abs/pii/S2451904924001021?via%3Dihub	Volume 49, March 2024, 102484
[6]	AI in HVAC fault detection and diagnosis: A systematic review Jian Bi [a], Hua Wang [a], Enbo Yan [a], Chuan Wang [a], Ke Yan [a b], Liangliang Jiang [c], Bin Yang [d] https://www.sciencedirect.com/science/article/pii/S277297022400004X?via%3Dihub	Volume 3, Issue 2, June 2024, 100071
[7]	Energy Efficiency and Human Comfort: AI and IoT Integration in Hospital HVAC Systems Shalom Akhai, Alex Khang	2024

	https://www.igi-global.com/chapter/energy-efficiency-and-human-comfort/%20341112	
[8]	Analyzing Machine Learning Algorithms applied to HVAC Systems for Sustainability and Efficiency Dubey Dhanraj Shevendrakumar, Shruti Maheshwari, Heeba Shaikh https://www.ijisae.org/index.php/IJISAE/article/view/3677	VOL. 12 NO. 1 (2024) / Research Article
[9]	Successful application of predictive information in deep reinforcement learning control: A case study based on an office building HVAC system Yuan Gao a, Shanrui Shi b, Shohei Miyata b, Yasunori Akashi b https://www.sciencedirect.com/science/article/abs/pii/S0360544224001154	Volume 291, 15 March 2024, 130344
[10]	ASSESSING THE IMPACT OF CLIMATE CHANGE ON HVAC SYSTEM DESIGN AND PROJECT MANAGEMENT https://fepbl.com/index.php/ijarss/article/view/848 Wisdom Ebirim,Favour Oluwadamilare Usman,Danny Jose Portillo Montero,Nwakamma Ninduwezuor-Ehiobu,Emmanuel Chigozie Ani, Kehinde Andrew Olu-lawal	Vol. 6 No. 3 (2024)
[11]	Improved energy management of chiller system with AI-based regression https://www.sciencedirect.com/science/article/abs/pii/S1568494623011092 Fu-Wing Yu, Wai-Tung Ho, Chak-Fung Jeff Wong	Volume 150, January 2024, 111091
[12]	Modelling building HVAC control strategies using a deep reinforcement learning approach.	Volume 310, 1 May

	https://www.sciencedirect.com/science/article/abs/pii/S0378778824001816?via%3Dihu Anh Tuan Nguyen [a], Duy Hoang Pham [a], Bee Lan Oo [b], Mattheos Santamouris [b], Yonghan Ahn [a], Benson T.H. Lim [b]	2024, 114065
[13]	https://weatherspark.com/	2024
[14]	https://www.siemens.com/global/en.html	2024
[15]	https://www.johnsoncontrols.com/	2014
[16]	https://www.se.com/no/no/	2014
[17]	https://www.bravida.no/	2014
[18]	https://openweathermap.org/	2024
[19]	Data Modeling Techniques for Data Warehousing Chuck Ballard, Dirk Herreman, Don Schau, Rhonda Bell, Eunsaeng Kim, Ann Valencic https://eddyswork.synthasite.com/resources/Data%20Modeling%20Tech%20For%20Data%20Warehouseing.pdf	February 1998
[20]	https://azure.microsoft.com/en-us/products/iot-hub	
[21]	https://www.postman.com/what-is-an-api/	
[22]	https://www.snowflake.com/trending/etl-tools/	
[23]	Pattern-Based Contextual Anomaly Detection in HVAC Systems Mohsin Munir; Andreas Dengel; Sheraz Ahmed https://www.dfki.de/fileadmin/user_upload/import/9505_pattern-based-contextual-anomaly-detection.pdf	November 2017
[24]	A comprehensive survey on design and application of autoencoder in deep learning; Pengzhi Li [a], Yan Pei [b], Jianqiang Li [c] https://www.sciencedirect.com/science/article/abs/pii/S1568494623001941	Volume 138, May 2023, 110176
[25]	https://www.dhs.wisconsin.gov/chemical/carbondioxide.htm	2024
[26]	Probabilistic Deep Learning Oliver Durr, Beate sick, Elvis Murina https://books.google.no/books?hl=no&lr=&id=-bYCEAAAQBAJ&oi=fnd&pg=PA1&dq=The+sequential+model+	2020 Page 25 to 62

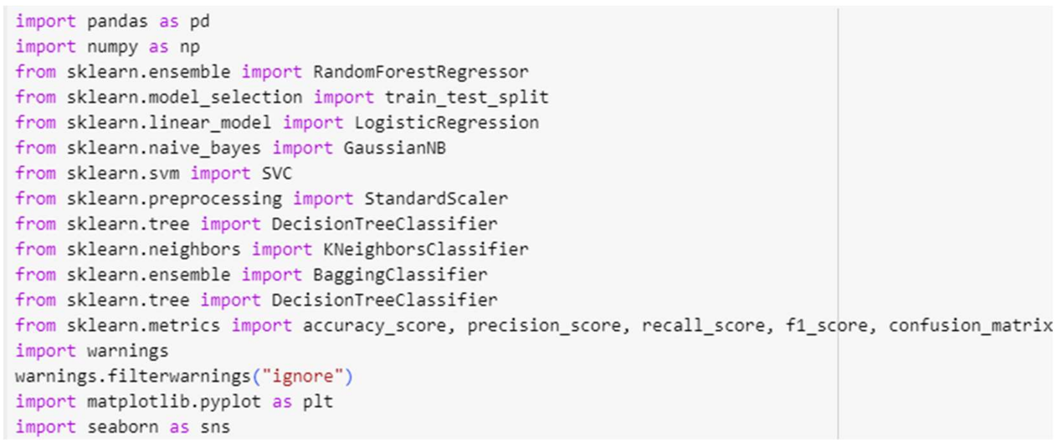
	with+tensorflow+and+keras&ots=xV7-Xn6XSy&sig=Vn4c4gyqKXX_Xhu4bs6TuRxAQdg&redir_esc=y#v=onepage&q=The%20sequential%20model%20with%20tensorflow%20and%20keras&f=false	
[27]	Multivariate time series clustering and forecasting for building energy analysis: Application to weather data quality control Luís Sanhudo a, João Rodrigues a, Ênio Vasconcelos Filho b https://www.sciencedirect.com/science/article/abs/pii/S2352710220336287?via%3Dihub	Volume 35, March 2021, 101996
[28]	An unsupervised data mining-based framework for evaluation and optimization of operation strategy of HVAC system Zhe Tian a, Zhonghui Lu a, Yakai Lu b, Qiang Zhang a, Xinyi Lin a , Jide Niu a https://www.sciencedirect.com/science/article/abs/pii/S0360544223034370?via%3Dihub	Volume 291, 15 March 2024, 130043
[29]	An unsupervised data mining-based framework for evaluation and optimization of operation strategy of HVAC system Zhe Tian a, Zhonghui Lu a, Yakai Lu b, Qiang Zhang a, Xinyi Lin a, Jide Niu a https://www.sciencedirect.com/science/article/abs/pii/S0360544223034370?via%3Dihub	March 15, 2024
[30]	Keras: The high-level API for TensorFlow tensorflow https://www.tensorflow.org/guide/keras	May 2024
[31]	Illustrated Guide to LSTM’s and GRU’s Michael Phi	September 18

	https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21	
[32]	https://azure.microsoft.com/en-us/solutions/ai	2024
[33]	https://www.microsoft.com/en-us/microsoft-fabric	2024
[34]	https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-are-machine-learning-algorithms	2024
[35]	https://learn.microsoft.com/en-us/azure/architecture/solution-ideas/articles/next-order-forecasting	2024
[36]	https://www.microsoft.com/en-us/sql-server	2024
[37]	https://www.postgresql.org/	2024
[38]	https://azure.microsoft.com/en-us/products/iot-hub	2024
[39]	https://learn.microsoft.com/en-us/azure/architecture/data-guide/relational-data/etl	2024

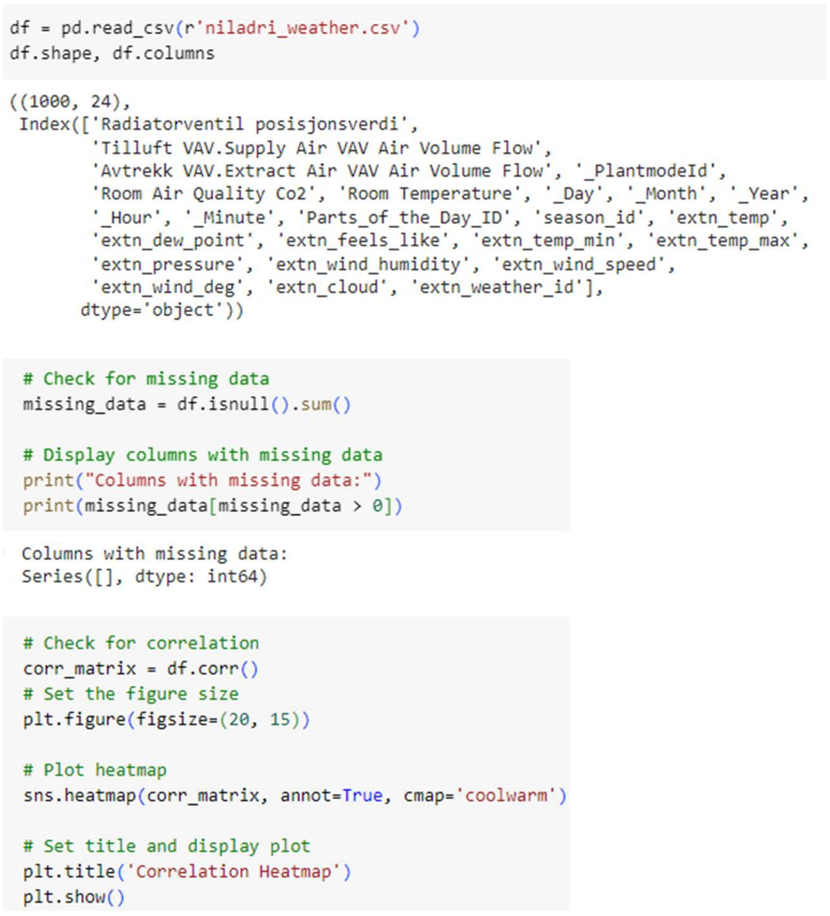
Appendix A

Sample Code

Figure 12: Heatmap



```
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix
import warnings
warnings.filterwarnings("ignore")
import matplotlib.pyplot as plt
import seaborn as sns
```



```
df = pd.read_csv(r'niladri_weather.csv')
df.shape, df.columns
```

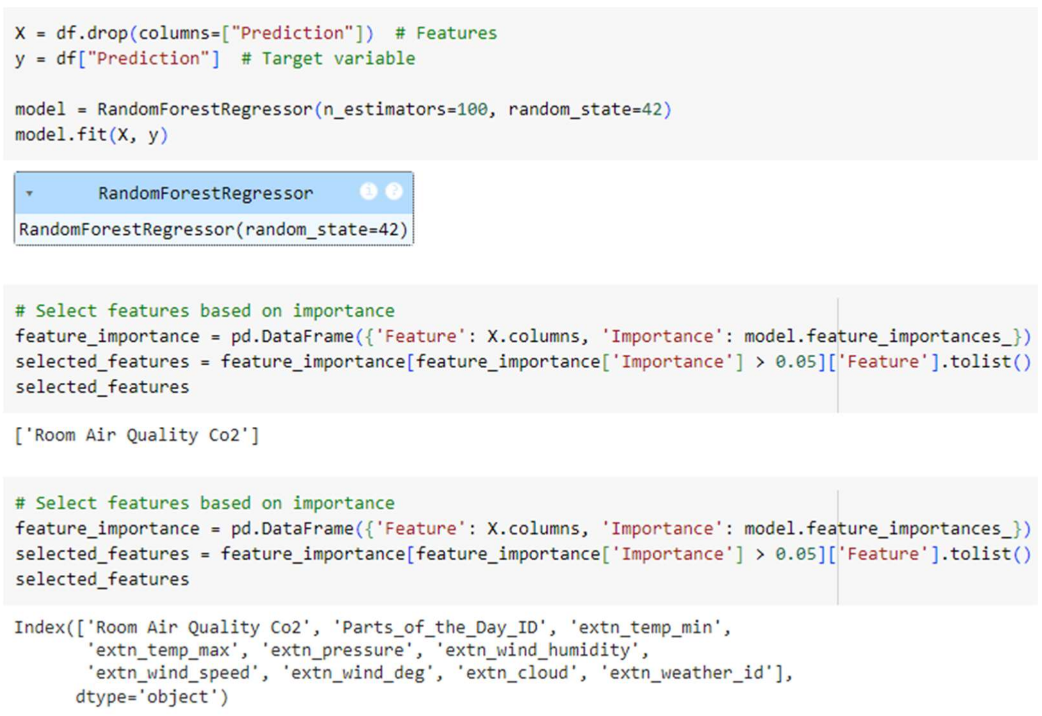
```
((1000, 24),
 Index(['Radiatorventil posisjonsverdi',
        'Tilluft VAV.Supply Air VAV Air Volume Flow',
        'Avtrekk VAV.Extract Air VAV Air Volume Flow', '_PlantmodeId',
        'Room Air Quality Co2', 'Room Temperature', '_Day', '_Month', '_Year',
        '_Hour', '_Minute', 'Parts_of_the_Day_ID', 'season_id', 'extn_temp',
        'extn_dew_point', 'extn_feels_like', 'extn_temp_min', 'extn_temp_max',
        'extn_pressure', 'extn_wind_humidity', 'extn_wind_speed',
        'extn_wind_deg', 'extn_cloud', 'extn_weather_id'],
       dtype='object'))
```

```
# Check for missing data
missing_data = df.isnull().sum()

# Display columns with missing data
print("Columns with missing data:")
print(missing_data[missing_data > 0])
```

```
Columns with missing data:
Series([], dtype: int64)
```

```
# Check for correlation
corr_matrix = df.corr()
# Set the figure size
plt.figure(figsize=(20, 15))

# Plot heatmap
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')

# Set title and display plot
plt.title('Correlation Heatmap')
plt.show()
```

Figure 16: Machine Learning Models



```
X = df.drop(columns=["Prediction"])  # Features
y = df["Prediction"]  # Target variable

model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X, y)
```

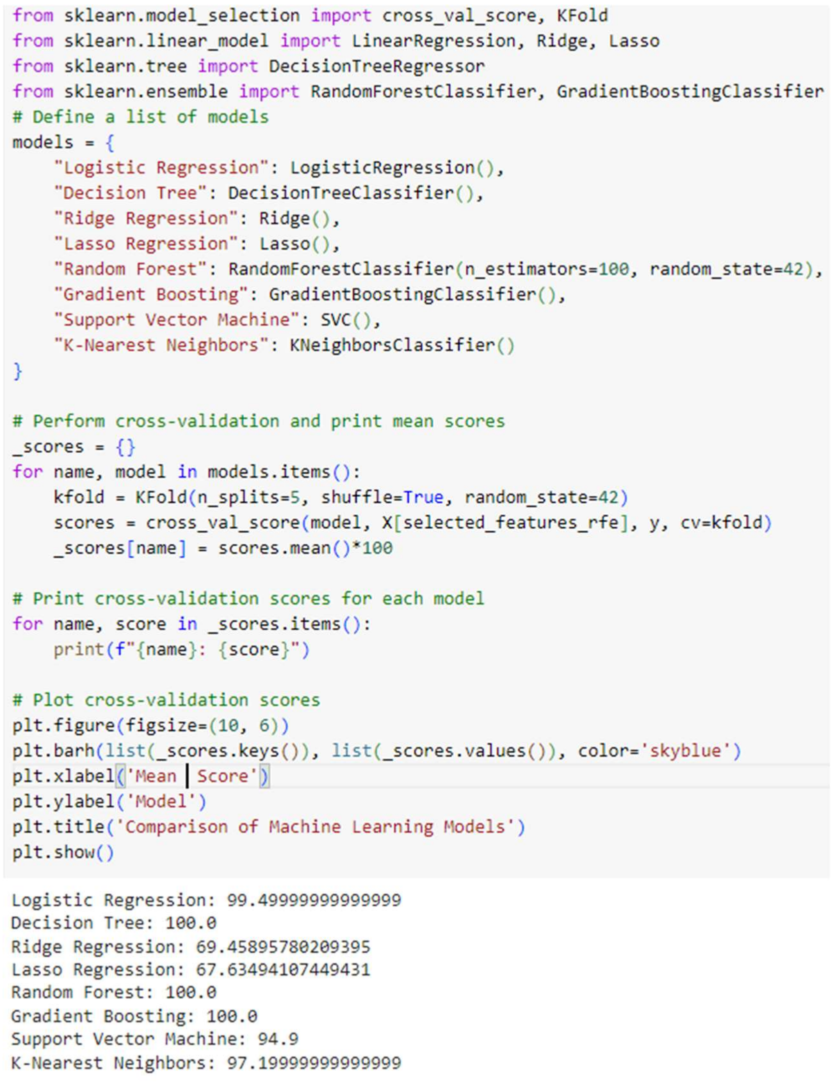
```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

```
# Select features based on importance
feature_importance = pd.DataFrame({'Feature': X.columns, 'Importance': model.feature_importances_})
selected_features = feature_importance[feature_importance['Importance'] > 0.05]['Feature'].tolist()
selected_features
```

```
['Room Air Quality Co2']
```

```
# Select features based on importance
feature_importance = pd.DataFrame({'Feature': X.columns, 'Importance': model.feature_importances_})
selected_features = feature_importance[feature_importance['Importance'] > 0.05]['Feature'].tolist()
selected_features
```

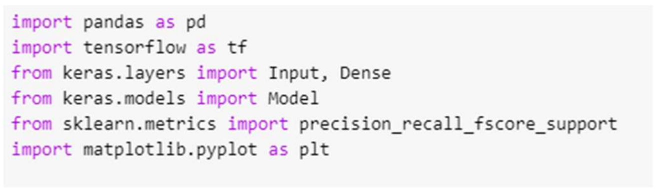
```
Index(['Room Air Quality Co2', 'Parts_of_the_Day_ID', 'extn_temp_min',
       'extn_temp_max', 'extn_pressure', 'extn_wind_humidity',
       'extn_wind_speed', 'extn_wind_deg', 'extn_cloud', 'extn_weather_id'],
      dtype='object')
```



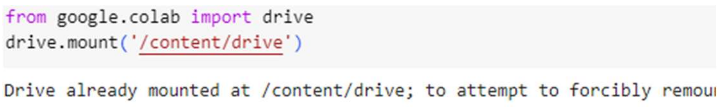
```
from sklearn.model_selection import cross_val_score, KFold
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
# Define a list of models
models = {
    "Logistic Regression": LogisticRegression(),
    "Decision Tree": DecisionTreeClassifier(),
    "Ridge Regression": Ridge(),
    "Lasso Regression": Lasso(),
    "Random Forest": RandomForestClassifier(n_estimators=100, random_state=42),
    "Gradient Boosting": GradientBoostingClassifier(),
    "Support Vector Machine": SVC(),
    "K-Nearest Neighbors": KNeighborsClassifier()
}

# Perform cross-validation and print mean scores
_scores = {}
for name, model in models.items():
    kfold = KFold(n_splits=5, shuffle=True, random_state=42)
    scores = cross_val_score(model, X[selected_features_rfe], y, cv=kfold)
    _scores[name] = scores.mean()*100

# Print cross-validation scores for each model
for name, score in _scores.items():
    print(f"{name}: {score}")

# Plot cross-validation scores
plt.figure(figsize=(10, 6))
plt.barh(list(_scores.keys()), list(_scores.values()), color='skyblue')
plt.xlabel('Mean Score')
plt.ylabel('Model')
plt.title('Comparison of Machine Learning Models')
plt.show()
```

```
Logistic Regression: 99.49999999999999
Decision Tree: 100.0
Ridge Regression: 69.45895780209395
Lasso Regression: 67.63494107449431
Random Forest: 100.0
Gradient Boosting: 100.0
Support Vector Machine: 94.9
K-Nearest Neighbors: 97.19999999999999
```

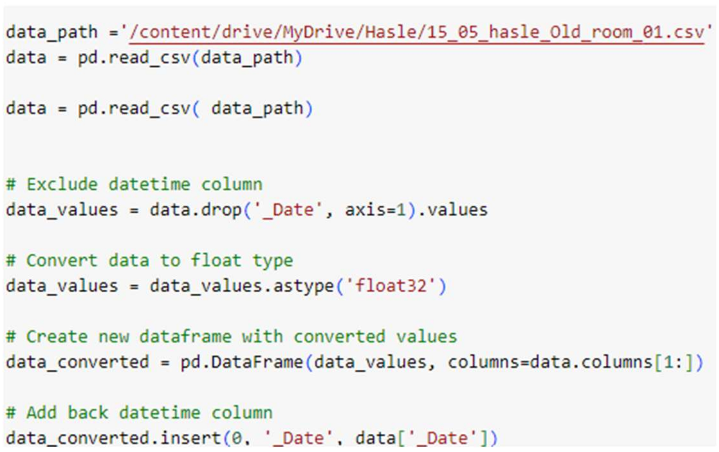
Figure 22: Autoencoder Anomaly detection on time series data



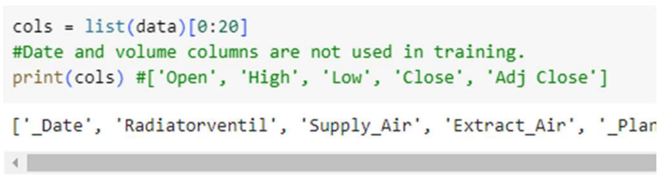
```
import pandas as pd
import tensorflow as tf
from keras.layers import Input, Dense
from keras.models import Model
from sklearn.metrics import precision_recall_fscore_support
import matplotlib.pyplot as plt
```



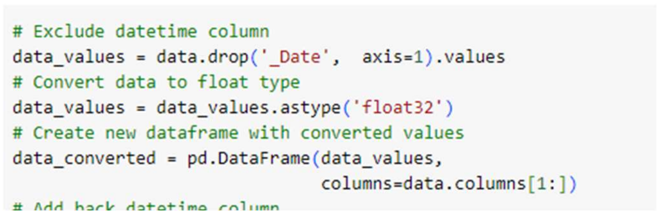
```
from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remou
```



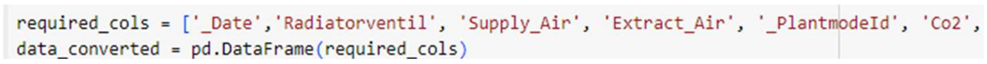
```
data_path ='/content/drive/MyDrive/Hasle/15_05_hasle_Old_room_01.csv'
data = pd.read_csv(data_path)

data = pd.read_csv( data_path)


# Exclude datetime column
data_values = data.drop('_Date', axis=1).values

# Convert data to float type
data_values = data_values.astype('float32')

# Create new dataframe with converted values
data_converted = pd.DataFrame(data_values, columns=data.columns[1:])

# Add back datetime column
data_converted.insert(0, '_Date', data['_Date'])
```



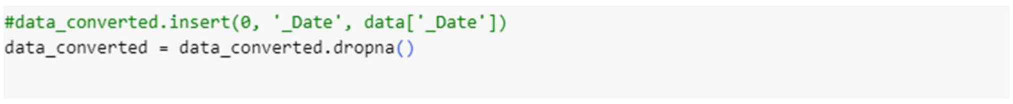
```
cols = list(data)[0:20]
#Date and volume columns are not used in training.
print(cols) #['Open', 'High', 'Low', 'Close', 'Adj Close']

['_Date', 'Radiatorventil', 'Supply_Air', 'Extract_Air', '_Plar
```



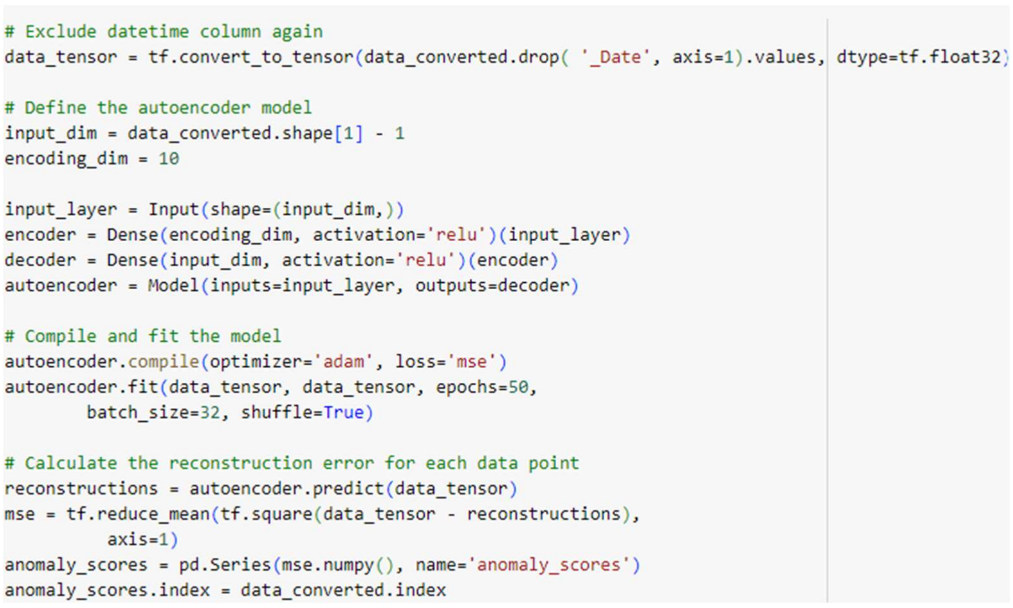
```
# Exclude datetime column
data_values = data.drop('_Date',  axis=1).values
# Convert data to float type
data_values = data_values.astype('float32')
# Create new dataframe with converted values
data_converted = pd.DataFrame(data_values,
                              columns=data.columns[1:])
# Add back datetime column
```



```
required_cols = ['_Date','Radiatorventil', 'Supply_Air', 'Extract_Air', '_PlantmodeId', 'Co2',
data_converted = pd.DataFrame(required_cols)
```



```
#data_converted.insert(0, '_Date', data['_Date'])
data_converted = data_converted.dropna()
```



```
# Exclude datetime column again
data_tensor = tf.convert_to_tensor(data_converted.drop( '_Date', axis=1).values, dtype=tf.float32)

# Define the autoencoder model
input_dim = data_converted.shape[1] - 1
encoding_dim = 10

input_layer = Input(shape=(input_dim,))
encoder = Dense(encoding_dim, activation='relu')(input_layer)
decoder = Dense(input_dim, activation='relu')(encoder)
autoencoder = Model(inputs=input_layer, outputs=decoder)

# Compile and fit the model
autoencoder.compile(optimizer='adam', loss='mse')
autoencoder.fit(data_tensor, data_tensor, epochs=50,
        batch_size=32, shuffle=True)

# Calculate the reconstruction error for each data point
reconstructions = autoencoder.predict(data_tensor)
mse = tf.reduce_mean(tf.square(data_tensor - reconstructions),
          axis=1)
anomaly_scores = pd.Series(mse.numpy(), name='anomaly_scores')
anomaly_scores.index = data_converted.index
```

```
print(anomaly_scores)
```

```
0        171334.218750
1        171263.265625
2        171300.906250
3        171195.281250
4        171232.609375
             ...
43476    171380.359375
43477    171380.953125
43478    171343.265625
43479    171300.718750
43480    171263.046875
Name: anomaly_scores, Length: 43481, dtype: float32
```

```
threshold = anomaly_scores.quantile(0.99)
anomalous = anomaly_scores > threshold
binary_labels = anomalous.astype(int)
f1_score = precision_recall_fscore_support(binary_labels, anomalous, average='binary')
```

```
test = data_converted['Co2'].values
predictions = anomaly_scores.values

print("Precision: ", precision)
print("Recall: ", recall)
print("F1 Score: ", f1_score)
```

```
Precision:  1.0
Recall:  1.0
F1 Score:  (1.0, 1.0, 1.0, None)
```

Figure 43: Temperature prediction with multiple time series columns 125 days.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
import tensorflow as tf
```

```
# set data index as datetime column
df.index = pd.to_datetime(df._Date)

# filter the columns by only the required_columns
required_cols = ['extn_temp', 'extn_pressure', 'extn_wind_speed', 'extn_wind_humidity', 'Supply_Air', 'Extract_Air', 'RoomTemperature', 'Co2', 'Radiatorventil']
df = df[required_cols]
df.head()
```

	extn_temp	extn_pressure	extn_wind_speed	extn_wind_humidity	Supply_Air	Extract_Air	RoomTemperature	Co2	Radiatorventil
_Date									
2022-01-31 02:30:00	-6.25	1008.0	2.75	54.0	0.0	0.0	19.9	402.0	0.0
2022-01-31 03:00:00	-6.20	1008.0	2.70	54.0	0.0	0.0	19.9	402.0	0.0
2022-01-31 03:30:00	-6.20	1008.0	2.70	54.0	0.0	0.0	19.9	402.0	0.0
2022-01-31 04:00:00	-6.07	1007.0	2.48	42.0	0.0	0.0	19.9	402.0	0.0
2022-01-31 04:30:00	-6.07	1007.0	2.48	42.0	0.0	0.0	19.8	402.0	0.0

```
# check number of nan values in dataframe
df.isna().sum()
```

```
extn_temp             0
extn_pressure         0
extn_wind_speed       0
extn_wind_humidity    0
Supply_Air            0
Extract_Air           0
RoomTemperature       0
Co2                   0
Radiatorventil        0
dtype: int64
```

```
df_final = df.resample('H').mean()
df_final.head()
```

```
df_final = df.resample('H').mean()
df_final.head()
```

_Date	extn_temp	extn_pressure	extn_wind_speed	extn_wind_humidity	Supply_Air	Extract_Air	RoomTemperature	Co2	Radiatorventil
2022-01-31 02:00:00	-6.25	1008.0	2.75	54.0	0.0	0.0	19.90	402.0	0.0
2022-01-31 03:00:00	-6.20	1008.0	2.70	54.0	0.0	0.0	19.90	402.0	0.0
2022-01-31 04:00:00	-6.07	1007.0	2.48	42.0	0.0	0.0	19.85	402.0	0.0
2022-01-31 05:00:00	-6.87	1007.0	2.42	31.0	0.0	0.0	19.80	402.0	0.0
2022-01-31 06:00:00	-6.88	1007.0	2.36	37.0	0.0	0.0	19.80	402.0	0.0

```
df_final.isna().sum()
```

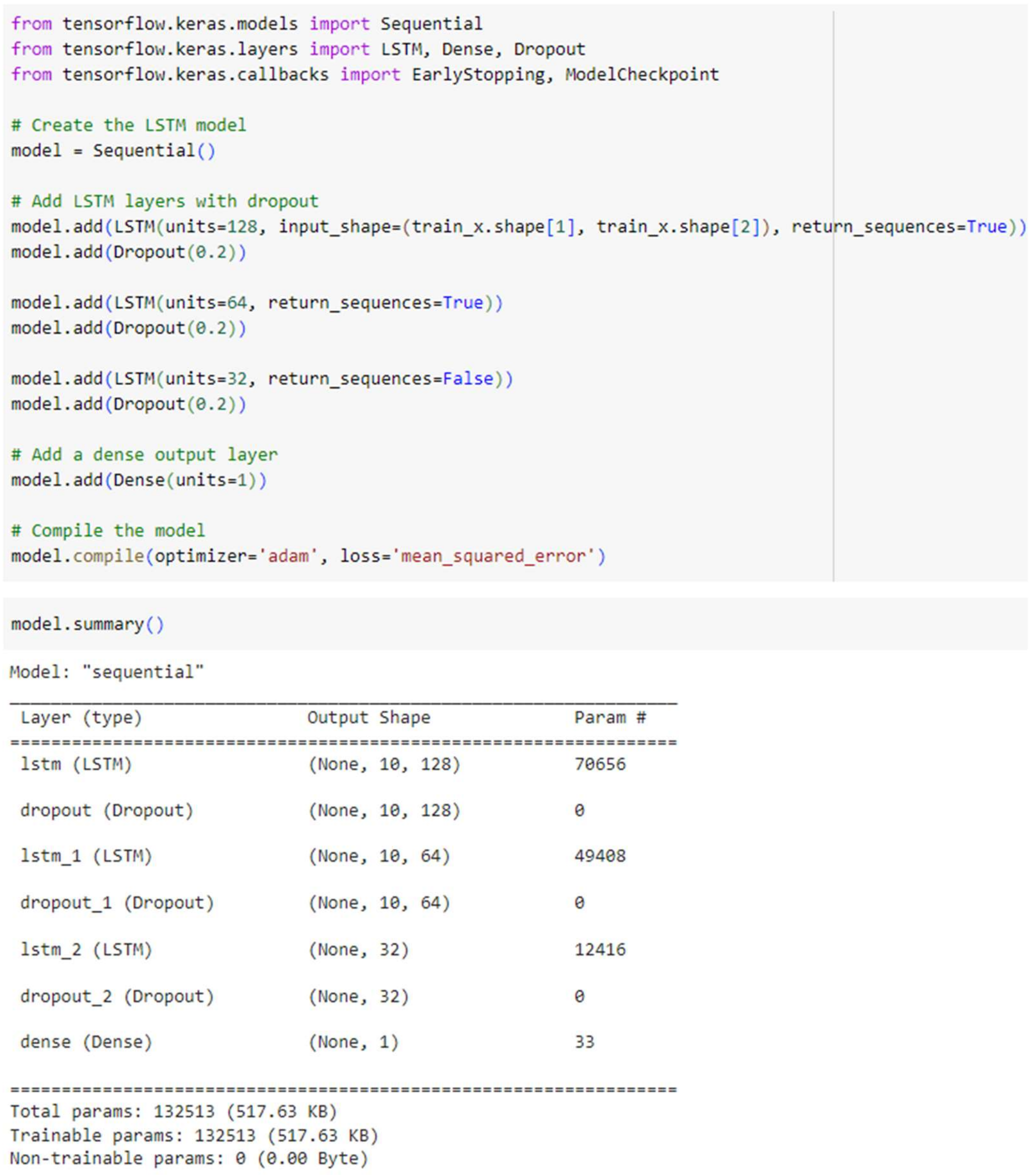
```
extn_temp             1632
extn_pressure         1632
extn_wind_speed       1632
extn_wind_humidity    1632
Supply_Air            1632
Extract_Air           1632
RoomTemperature       1632
Co2                   1632
Radiatorventil        1632
dtype: int64
```

```
df_final = df_final.fillna(method='ffill')
df_final.isna().sum()
```

```
extn_temp             0
extn_pressure         0
extn_wind_speed       0
extn_wind_humidity    0
Supply_Air            0
Extract_Air           0
RoomTemperature       0
Co2                   0
Radiatorventil        0
dtype: int64
```

```
from sklearn.preprocessing import MinMaxScaler

# Normalize the data
scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(df_final)

# Define sequence length and features
sequence_length = 10  # Number of time steps in each sequence
num_features = len(df_final.columns)

# Create sequences and corresponding labels
sequences = []
labels = []
for i in range(len(scaled_data) - sequence_length):
    seq = scaled_data[i:i+sequence_length]
    label = scaled_data[i+sequence_length][6]  # '_tempm' column index
    #label = scaled_data[i+sequence_length][0]  # '_tempm' column index
    sequences.append(seq)
    labels.append(label)

# Convert to numpy arrays
sequences = np.array(sequences)
labels = np.array(labels)

# Split into train and test sets
train_size = int(0.8 * len(sequences))
train_x, test_x = sequences[:train_size], sequences[train_size:]
train_y, test_y = labels[:train_size], labels[train_size:]

print("Train X shape:", train_x.shape)
print("Train Y shape:", train_y.shape)
print("Test X shape:", test_x.shape)
print("Test Y shape:", test_y.shape)
```

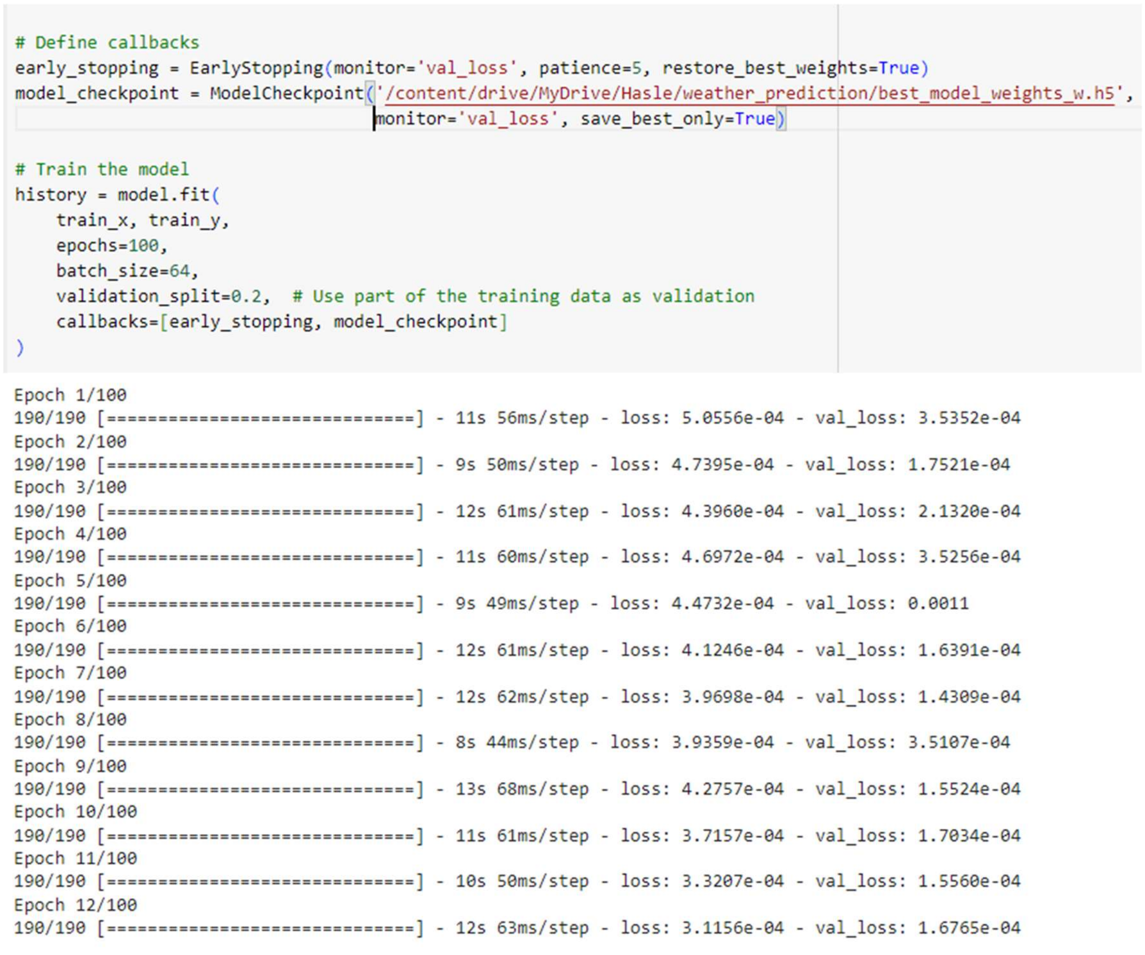
```
Train X shape: (15177, 10, 9)
Train Y shape: (15177,)
Test X shape: (3795, 10, 9)
Test Y shape: (3795,)
```



```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint

# Create the LSTM model
model = Sequential()

# Add LSTM layers with dropout
model.add(LSTM(units=128, input_shape=(train_x.shape[1], train_x.shape[2]), return_sequences=True))
model.add(Dropout(0.2))

model.add(LSTM(units=64, return_sequences=True))
model.add(Dropout(0.2))

model.add(LSTM(units=32, return_sequences=False))
model.add(Dropout(0.2))

# Add a dense output layer
model.add(Dense(units=1))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')
```

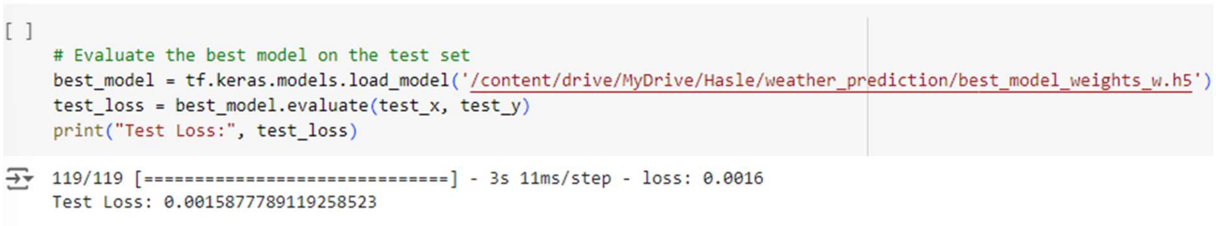
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 10, 128)           70656

 dropout (Dropout)           (None, 10, 128)           0

 lstm_1 (LSTM)               (None, 10, 64)            49408

 dropout_1 (Dropout)         (None, 10, 64)            0

 lstm_2 (LSTM)               (None, 32)                12416

 dropout_2 (Dropout)         (None, 32)                0

 dense (Dense)               (None, 1)                 33

=================================================================
Total params: 132513 (517.63 KB)
Trainable params: 132513 (517.63 KB)
Non-trainable params: 0 (0.00 Byte)
```



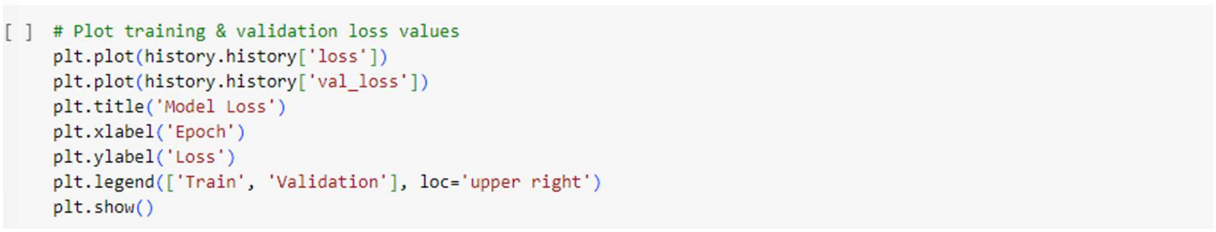
```
# Define callbacks
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
model_checkpoint = ModelCheckpoint('/content/drive/MyDrive/Hasle/weather_prediction/best_model_weights_w.h5',
                                   monitor='val_loss', save_best_only=True)

# Train the model
history = model.fit(
    train_x, train_y,
    epochs=100,
    batch_size=64,
    validation_split=0.2,  # Use part of the training data as validation
    callbacks=[early_stopping, model_checkpoint]
)
```

```
Epoch 1/100
190/190 [==============================] - 11s 56ms/step - loss: 5.0556e-04 - val_loss: 3.5352e-04
Epoch 2/100
190/190 [==============================] - 9s 50ms/step - loss: 4.7395e-04 - val_loss: 1.7521e-04
Epoch 3/100
190/190 [==============================] - 12s 61ms/step - loss: 4.3960e-04 - val_loss: 2.1320e-04
Epoch 4/100
190/190 [==============================] - 11s 60ms/step - loss: 4.6972e-04 - val_loss: 3.5256e-04
Epoch 5/100
190/190 [==============================] - 9s 49ms/step - loss: 4.4732e-04 - val_loss: 0.0011
Epoch 6/100
190/190 [==============================] - 12s 61ms/step - loss: 4.1246e-04 - val_loss: 1.6391e-04
Epoch 7/100
190/190 [==============================] - 12s 62ms/step - loss: 3.9698e-04 - val_loss: 1.4309e-04
Epoch 8/100
190/190 [==============================] - 8s 44ms/step - loss: 3.9359e-04 - val_loss: 3.5107e-04
Epoch 9/100
190/190 [==============================] - 13s 68ms/step - loss: 4.2757e-04 - val_loss: 1.5524e-04
Epoch 10/100
190/190 [==============================] - 11s 61ms/step - loss: 3.7157e-04 - val_loss: 1.7034e-04
Epoch 11/100
190/190 [==============================] - 10s 50ms/step - loss: 3.3207e-04 - val_loss: 1.5560e-04
Epoch 12/100
190/190 [==============================] - 12s 63ms/step - loss: 3.1156e-04 - val_loss: 1.6765e-04
```

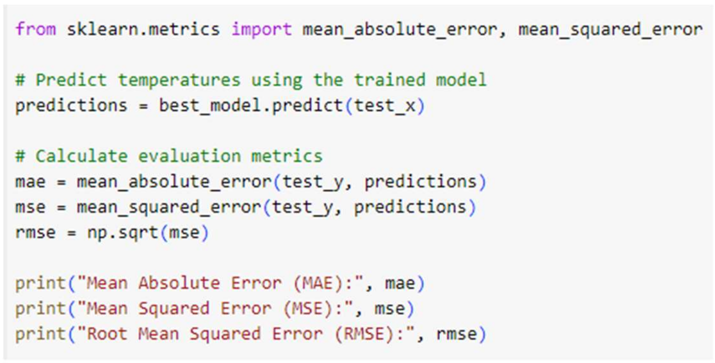


```
[ ]
    # Evaluate the best model on the test set
    best_model = tf.keras.models.load_model('/content/drive/MyDrive/Hasle/weather_prediction/best_model_weights_w.h5')
    test_loss = best_model.evaluate(test_x, test_y)
    print("Test Loss:", test_loss)
```

```
119/119 [==============================] - 3s 11ms/step - loss: 0.0016
Test Loss: 0.0015877789119258523
```

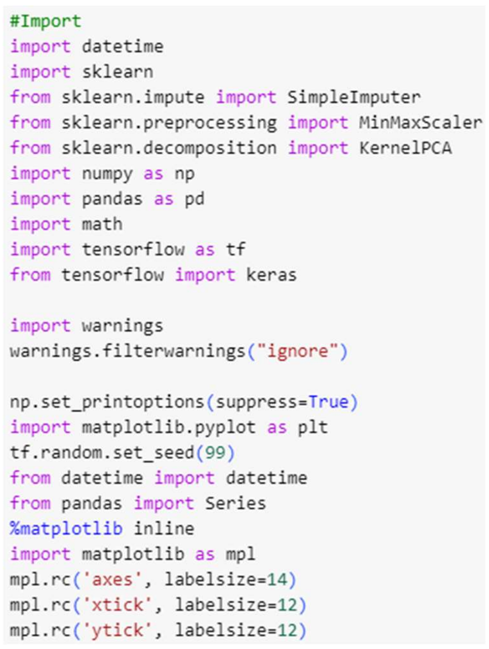


```
[ ] # Plot training & validation loss values
    plt.plot(history.history['loss'])
    plt.plot(history.history['val_loss'])
    plt.title('Model Loss')
    plt.xlabel('Epoch')
    plt.ylabel('Loss')
    plt.legend(['Train', 'Validation'], loc='upper right')
    plt.show()
```

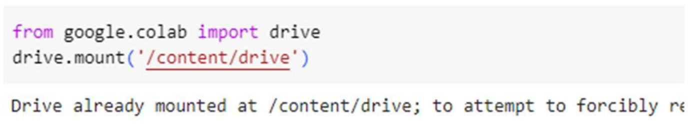


```
from sklearn.metrics import mean_absolute_error, mean_squared_error

# Predict temperatures using the trained model
predictions = best_model.predict(test_x)

# Calculate evaluation metrics
mae = mean_absolute_error(test_y, predictions)
mse = mean_squared_error(test_y, predictions)
rmse = np.sqrt(mse)

print("Mean Absolute Error (MAE):", mae)
print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
```

Figure 53: Air supply and Co2 predicted with another co- related variables.

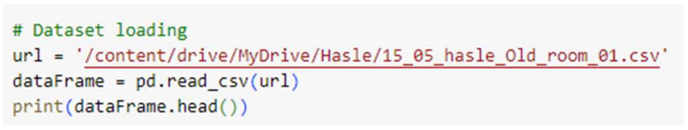


```
#Import
import datetime
import sklearn
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import MinMaxScaler
from sklearn.decomposition import KernelPCA
import numpy as np
import pandas as pd
import math
import tensorflow as tf
from tensorflow import keras

import warnings
warnings.filterwarnings("ignore")

np.set_printoptions(suppress=True)
import matplotlib.pyplot as plt
tf.random.set_seed(99)
from datetime import datetime
from pandas import Series
%matplotlib inline
import matplotlib as mpl
mpl.rc('axes', labelsize=14)
mpl.rc('xtick', labelsize=12)
mpl.rc('ytick', labelsize=12)
```

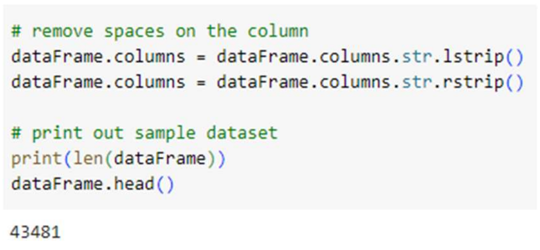


```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly re
```

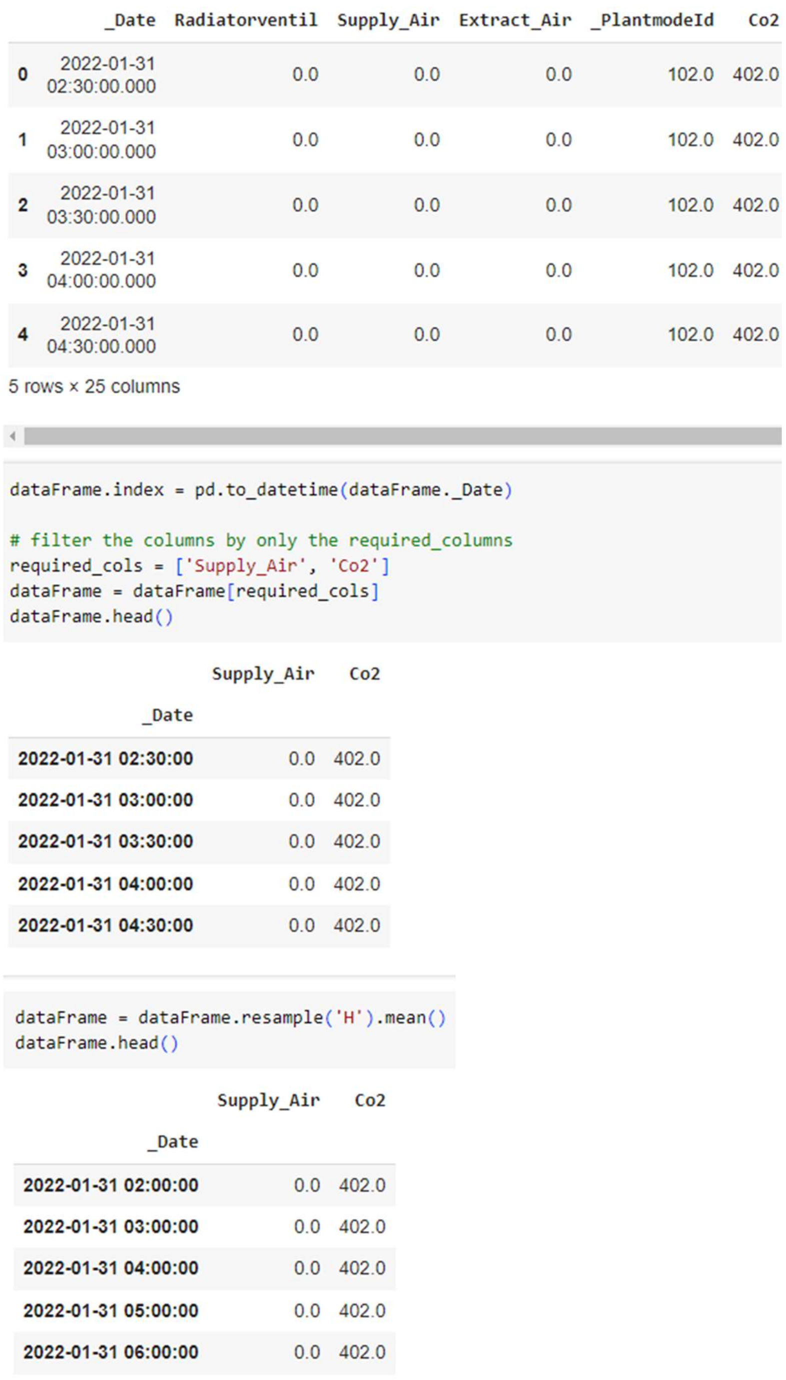


```
# Dataset loading
url = '/content/drive/MyDrive/Hasle/15_05_hasle_Old_room_01.csv'
dataFrame = pd.read_csv(url)
print(dataFrame.head())
```



```
# remove spaces on the column
dataFrame.columns = dataFrame.columns.str.lstrip()
dataFrame.columns = dataFrame.columns.str.rstrip()

# print out sample dataset
print(len(dataFrame))
dataFrame.head()
```

```
43481
```



	_Date	Radiatorventil	Supply_Air	Extract_Air	_PlantmodeId	Co2
0	2022-01-31 02:30:00.000	0.0	0.0	0.0	102.0	402.0
1	2022-01-31 03:00:00.000	0.0	0.0	0.0	102.0	402.0
2	2022-01-31 03:30:00.000	0.0	0.0	0.0	102.0	402.0
3	2022-01-31 04:00:00.000	0.0	0.0	0.0	102.0	402.0
4	2022-01-31 04:30:00.000	0.0	0.0	0.0	102.0	402.0

5 rows × 25 columns

```
dataFrame.index = pd.to_datetime(dataFrame._Date)

# filter the columns by only the required_columns
required_cols = ['Supply_Air', 'Co2']
dataFrame = dataFrame[required_cols]
dataFrame.head()
```

	Supply_Air	Co2
_Date		
2022-01-31 02:30:00	0.0	402.0
2022-01-31 03:00:00	0.0	402.0
2022-01-31 03:30:00	0.0	402.0
2022-01-31 04:00:00	0.0	402.0
2022-01-31 04:30:00	0.0	402.0

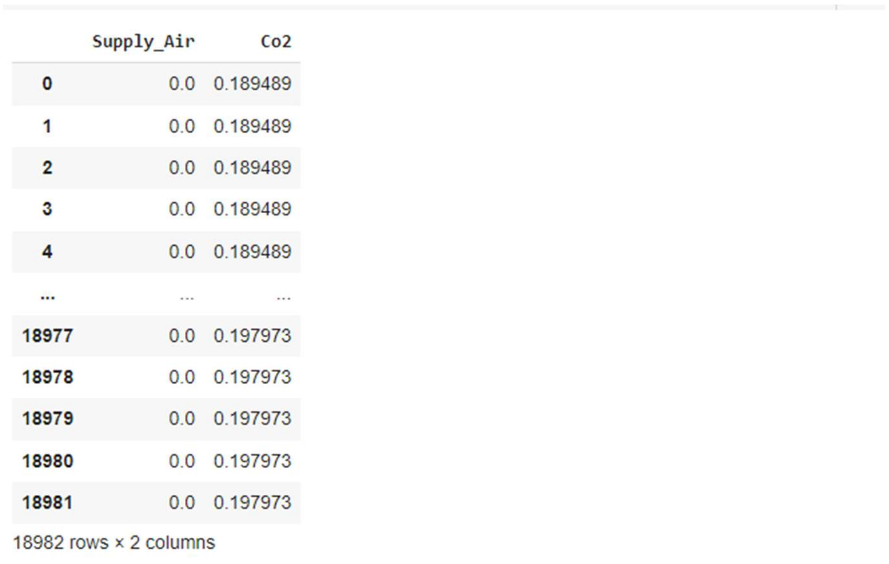
```
dataFrame = dataFrame.resample('H').mean()
dataFrame.head()
```

	Supply_Air	Co2
_Date		
2022-01-31 02:00:00	0.0	402.0
2022-01-31 03:00:00	0.0	402.0
2022-01-31 04:00:00	0.0	402.0
2022-01-31 05:00:00	0.0	402.0
2022-01-31 06:00:00	0.0	402.0

```
imputer = SimpleImputer(missing_values=np.nan)
#dataFrame.drop(columns=['_Date'], inplace=True)
dataFrame = pd.DataFrame(imputer.fit_transform(
  dataFrame), columns=dataFrame.columns)
dataFrame = dataFrame.reset_index(drop=True)
# Applying feature scaling
scaler = MinMaxScaler(feature_range=(0, 1))
df_scaled = scaler.fit_transform(dataFrame.to_numpy())
df_scaled = pd.DataFrame(df_scaled, columns=list(dataFrame.columns))

target_scaler = MinMaxScaler(feature_range=(0, 1))

df_scaled[['Supply_Air', 'Co2']] = target_scaler.fit_transform(
  dataFrame[['Supply_Air', 'Co2']].to_numpy())

df_scaled = df_scaled.astype(float)
print(df_scaled.head())
```

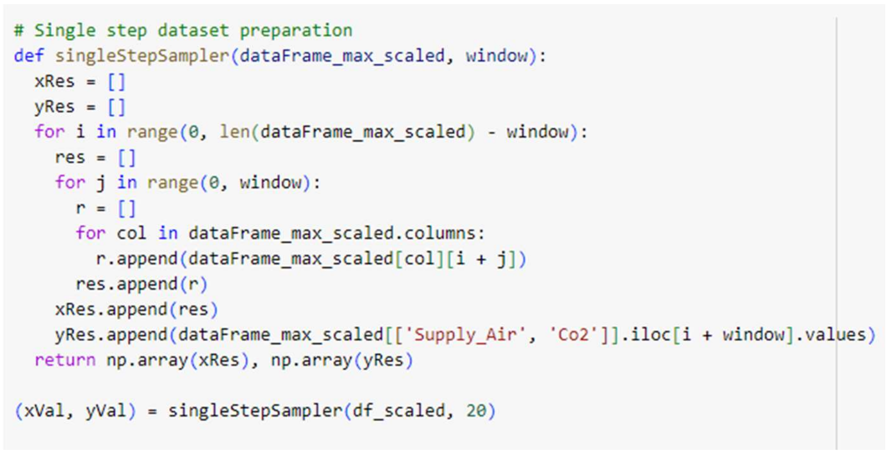
```
   Supply_Air       Co2
0         0.0  0.189489
1         0.0  0.189489
2         0.0  0.189489
3         0.0  0.189489
4         0.0  0.189489
```

```
# copy the data
dataFrame_max_scaled = df_scaled.copy()

# apply normalization techniques
for column in dataFrame_max_scaled.columns:
  dataFrame_max_scaled[column] = dataFrame_max_scaled[column] / dataFrame_max_scaled[column].abs().max()

# view normalized data
display(dataFrame_max_scaled)
```



	Supply_Air	Co2
0	0.0	0.189489
1	0.0	0.189489
2	0.0	0.189489
3	0.0	0.189489
4	0.0	0.189489
...	...	...
18977	0.0	0.197973
18978	0.0	0.197973
18979	0.0	0.197973
18980	0.0	0.197973
18981	0.0	0.197973

18982 rows × 2 columns



```
# Single step dataset preparation
def singleStepSampler(dataFrame_max_scaled, window):
  xRes = []
  yRes = []
  for i in range(0, len(dataFrame_max_scaled) - window):
    res = []
    for j in range(0, window):
      r = []
      for col in dataFrame_max_scaled.columns:
        r.append(dataFrame_max_scaled[col][i + j])
      res.append(r)
    xRes.append(res)
    yRes.append(dataFrame_max_scaled[['Supply_Air', 'Co2']].iloc[i + window].values)
  return np.array(xRes), np.array(yRes)

(xVal, yVal) = singleStepSampler(df_scaled, 20)
```

```
# Dataset splitting  *** df = dataFrame_max_scaled
SPLIT = 0.85

X_train = xVal[:int(SPLIT * len(xVal))]
y_train = yVal[:int(SPLIT * len(yVal))]
X_test = xVal[int(SPLIT * len(xVal)):]
y_test = yVal[int(SPLIT * len(yVal)):]
```

```
multivariate_gru = tf.keras.Sequential()
multivariate_gru.add(
  tf.keras.layers.GRU(200, input_shape=(X_train.shape[1], X_train.shape[2])))
multivariate_gru.add(
  tf.keras.layers.Dropout(0.5))

# Output layer for two predictor variables
multivariate_gru.add(
  tf.keras.layers.Dense(2, activation='linear'))

# Compile the model
multivariate_gru.compile(loss='MeanSquaredError',
            metrics=['MAE', 'MSE'],
            optimizer=tf.keras.optimizers.Adam())
multivariate_gru.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 gru_2 (GRU)                 (None, 200)               122400

 dropout_2 (Dropout)         (None, 200)               0

 dense_2 (Dense)             (None, 2)                 402

=================================================================
Total params: 122802 (479.70 KB)
Trainable params: 122802 (479.70 KB)
Non-trainable params: 0 (0.00 Byte)
```

```
history = multivariate_gru.fit(X_train, y_train, epochs=20)
```

```
predicted_values = multivariate_gru.predict(X_test)

d = {
  'Predicted_Supply_Air': predicted_values[:, 0],
  'Predicted_Co2': predicted_values[:, 1],
  'Actual_Supply_Air': y_test[:, 0],
  'Actual_Co2': y_test[:, 1],
}

d = pd.DataFrame(d)

fig, ax = plt.subplots(figsize=(10, 6))
plt.plot(d[['Actual_Supply_Air', 'Predicted_Supply_Air']], label=['Actual_Supply_Air', 'Predicted_Supply_Air'])
plt.plot(d[['Actual_Co2', 'Predicted_Co2']], label=['Actual_Co2', 'Predicted_Co2'])
plt.xlabel('Timestamps')
plt.ylabel('Values')
ax.legend()
plt.show()
```