Machine Learning Modelling for HVAC Systems

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Bachelor's thesis

Valkeakoski. Electrical and Automation Engineering

Spring 2021

Genrikh Ekkerman

HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Electrical and Automation Engineering Abstract

Author Genrikh Ekkerman Year 2021
Subject Machine Learning Modelling for HVAC Systems
Supervisors Juhani Henttonen

Vuores is a school building with the HVAC system deployed as part of its building automation system. Indoor conditions are maintained by ventilation and heating systems, which consume a major part of the total energy consumption of the building.

For solving this problem linear regression models were developed for the key parameters of the HVAC system. Models included battery network energy, for future optimization of energy consumption, air temperature and CO2 concentration models for future forecasting Indoor air quality parameters.

For developing these models, statistical and machine learning methods were used. Features for each model were selected using a repeated KFold cross-validation method. Fitted models were successfully evaluated on the unseen data, from the same source.

Keywords HVAC, machine learning, regression
Pages 27

Contents

Appendices

1 INTRODUCTION

Vuores is a school building located in Tampere, southern Finland. The Construction of the site was completed in August 2013. As of 2020, the school teaches 640 students. There are plans to expand the school's capacity to a thousand students in a few years. (Vuoreksen koulu, 2020)

HVAC or Heating Ventilation Air Conditioning systems are a major contributor to power consumption in European countries. A great opportunity arises for energy savings, due to many HVAC systems being not properly maintained nor optimized. (Knight, 2012, 6). Such systems are important for maintaining proper IAQ (indoor air quality) in the human occupied rooms in the building. Optimal control of HVAC can lead to significant resources spent reduction. For deployment of such control, it is important to have a model of the building's behaviour.

In the age of data gathering, Machine Learning has been used to optimize solving problems such as: classification, product recommendations, image & video recognition, and forecasting. The goal of this thesis was to apply Machine Learning for predicting parameters of the building maintained by a HVAC system. Practical part of the project focuses on challenges faced when processing building automation data and training optimal model forecasting important parameters.

2 THEORY

2.1 HVAC

2.1.1 HVAC system components

A Heating Ventilation and Air Conditioning (or HVAC) system is a group of components working on maintaining desired air conditions indoors. (Sugarman, 2005, 1-14)
Typical main HVAC components are:

- Heating part
  - Boiler for heating up water or steam in the system. Boilers are typically fuelled with electricity or natural gas.
  - Furnace for heating up air. Natural gas or electricity are main fuelling options.
  - Electrical Heating Coils for heating air.
- Ventilation part
  - MUA (Make-up air) exhaust systems like kitchen hoods, bathroom ventilation.
  - Air pressure controlling systems.
- Air Conditioning part

- Chilled water and refrigeration systems for removing heat from the air.
- Air temperature control systems
- Humidity control systems
- Air filtering systems
- Air velocity, volume, and direction of airflow controls
- Outside systems for processing incoming outdoors air such as: supply and return air ducts, fans, air inlets and outlets.

Figure 1. Inside looks of the HVAC system. (mindmingles, 2020)

2.1.2 HVAC system operation

Before the HVAC system starts working first it is important to figure out ventilation and IAQ requirements, depending on the size and specification of the ventilated area. Main goal is to cycle enough air to prevent explicit carbon dioxide making air “stale” and unbreathable. Carbon Dioxide is usually removed by replacing inside air with “fresh” outdoors air. Another import factor is to maintain proper positive pressurization of the environment. The usual process inside AHU (air handling unit) goes as follows, RE (return air, from the rooms to the system) goes into a mixed air-chamber, known as plenum. Then the RE is combined with OA (outside air, coming from outside air dampers) inside the plenum. The amount of OA depends on total volume of SA (supply air, going back to the rooms from the system) and should be 20% of SA, with this situation RE will comprise 80% of MA (mixed air in plenum) for a total of 100% MA. Then air leaves plenum and proceeds further into a coil section. Any air EA (exhaust air, brought to the system) is excluded from the process through exhaust air dampers. The process is detailed with arbitrary numbers in Figure 2.

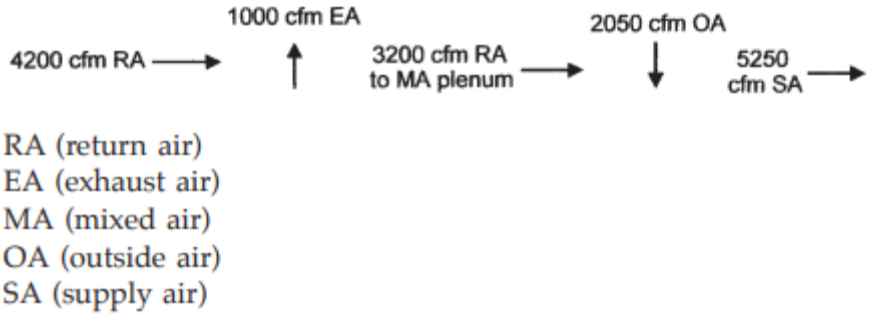

Figure 2. Air flow diagram of the ventilation process. (Sugarman, 2005)

The amount of heating is usually decided by indoor design features (such as, quality of insulation, number of doors and windows, area of the roof etc.) and outdoors weather conditions. For example, after leaving the coil section air carries X Wh of heat. It is blown by a supply air fan into an insulated supply air duct and through supply air outlets into the conditioned room. There, air gives up X Wh of heat to the environment, to replace X Wh of heat that left the room through walls, windows, doors ceiling, roof etc. After that air goes to RA inlets of AHU and the process is repeated. Similar logic applies for cooling, but air is cooled. In case of cooling, air is heated inside conditioned space by humans, electric appliances, and other heat sources. The example of the process is shown in Figure 3.

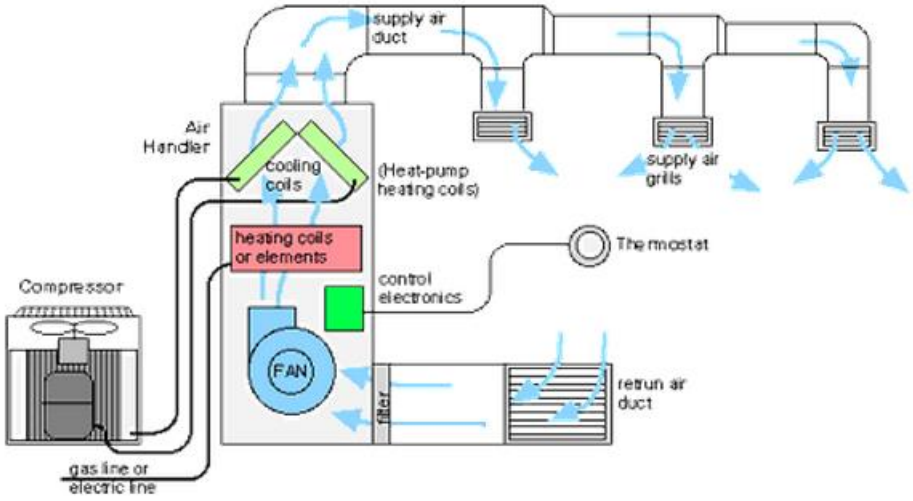

Figure 3. Schematics of HVAC system process. (RAY, 2015)

2.1.3 HVAC system control

There are three main types of HVAC control systems: pneumatic, electric, and electronic - Direct Digital Control (DDC, for short). DDC is the automated control of the process by a digital computer. A DDC control's loop main component is a microprocessor. Microprocessor's output is converted to pressure signals, to control cooling and heating valves of the system. DDC allows for better control of water temperature, supply air temperature and parameters of the ventilation system.

A combination of software and hardware solutions is used for maintaining specific variables in the desired condition. For example, logic on how indoors temperature should be conditioned, based on outdoors temperature can be assigned to a microprocessor. Logic of the DDC control system can be modified on the software level. This allows for the development of mathematical control algorithms. With smart control logic it allows for improved energy management and demand-based cooling and heating control. (Sugarman, 2005)

2.1.4 Air Quality Standards

According to Finnish classification of indoor air quality and thermal conditions, there are three categories of indoor climate: S1, S2 and S3. Air quality and thermal conditions are categorised based on values of target parameters. Those categories reflect on satisfaction and health risk from the indoor climate. In detail:

S1: Individual Indoor Climate. Depicts a climate of high quality with thermal conditions being comfortable and healthy throughout the whole year. This category should also satisfy people with specific requirements such as: people with respiratory illnesses or allergies and senior citizens.

S2: Good Indoor Climate. Depicts good indoor air quality with good temperature and amount of moisture in the air. But temperature may occasionally rise above comfort levels during hottest days in summertime.

S3: Satisfactory indoor Climate. Indoor air quality generally stays within limits set by building code. Sometimes air may feel dry and stale. Temperature rises above comfort levels during hot days in summertime. (Säteri, 2002)

Details of target parameters for thermal conditions can be seen in Table 1, target values for indoor air quality can be seen in Table 2.

		Unit	Indoor Climate Category Maximum values			Note
			S1	S2	S3	
Room temperature*	Winter	°C	(21-22)*	20-22	20-23	(I)
	Summer		(23-24)*	23-26	22-27(35)	***
Air velocity	Winter (20 °C)	m/s	0.13	0.16	0.19	(II)
	Winter (21 °C)	m/s	0.14	0.17	0.20	
Air velocity	Summer (24°C)	m/s	0.20	0.25	0.30	(II)

Table 1. Examples of target values for thermal conditions. Notes can be seen in appendix 1. (Säteri, 2002)

		Unit	Indoor climate category Maximum values			Note
			S1	S2	S3	
Radon	Rn	Bq/m³	100	100	200	(I)
Carbon dioxide	CO_2	ppm	700	900	1200	(II)
Carbon dioxide	CO_2	mg/m³	1300	1650	2200	
Ammonia and amines	NH_3	µg/m³	30	30	40	(III)
Formaldehyde	H_2CO	µg/m³	30	50	100	(IV)
Volatile organic compounds	TVOC	µg/m³	200	300	600	(V)
Carbon monoxide	CO	mg/m³	2	3	8	(VI)
Ozone	O_3	µg/m³	20	50	80	(VII)
Odor intensity (intensity scale)		-	3	4	5,5	(VIII)
Microbes			No maximum value			(IX)
Cigarette smoke in rooms for non-smokers			Not discernible			(X)
Mass concentration of airborne particulate matter	PM_{10}	µg/m³	20	40	50	(XI)

Table 2. Example of target values for indoor air quality. Notes can be seen in appendix 1. (Säteri, 2002)

Two important parameters in this According to Finnish classification of indoor air quality and thermal conditions, there are three categories of indoor climate standard are air temperature and CO2 concentration.

This standard is applicable for monitoring HVAC processes.

2.2 Machine Learning

Machine learning can be broadly defined as computational methods using experience to improve performance or to make accurate predictions. (Mohri, Rostamizadeh, & Talwalkar, 2018) In this case *experience* means past information provided for the learner, usually takes the form of electronic data collected and processed for analysis.

Machine Learning (or ML for short) finds its use when conventional computer programs are powerless. Main areas of application are classification, regression, image recognition, natural language processing, customer recommendations. Being trained on gathered sets of data, ML models can be later used to process unseen cases. A general problem for ML algorithms is one, where you need to find relation between an object which could be a numerical value, a set of pixels, a signal etc. and some response, which could be class, shape, value, model will learn over time relation between pair object-response and will try to approximate the function between those two. After such an approximation, the model should be able to generate reasonable responses from new, but related objects.

2.2.1 Supervised Learning

Supervised learning is a learning scenario, when the learner receives a training set consisting of labelled data. Data is usually split in training and testing sets, after learning, model to assess accuracy of the model, predicted label from test data is compared to real label. Special metrics are used for giving accuracy score for a model. Metrics will be discussed in detail in Chapter 2.5. Regression and classification are the most common use cases for supervised learning. (Mohri, Rostamizadeh, & Talwalkar, 2018)

Common types of input data are:

- Numerical features. List or lists of numerical values describing an object. This is the common type of problem.
- Time series. Object described with sequence if values. Time is often used as a parameter.
- Image or video. Image is represented with a list or matrix of numerical values, and the video is a sequence of the former.
- More rare cases have input data as distance matrix, text, or graphs.

Common types of outputs are:

- Set of answers represents the future state of the sequence, in this case it is a forecasting problem.
- Answers can be a value in a finite range, in this case it is a classification problem.

If the answer can be a value in an infinite range, then it is a regression problem.

2.2.2 Unsupervised Learning

Unsupervised learning is another common learning scenario, when the learner exclusively receives a set of unlabelled data. (Mohri, Rostamizadeh, & Talwalkar, 2018)Unlike in supervised learning there are no assessment metrics used since there is usually no definite correct answer. Some example problems include clustering, dimensionality reduction, anomaly detection and association mining.

Input datasets could be easier to compose, since no labels are required, but unsupervised learning is applied to problems with large amounts of features. A major disadvantage is the inability to evaluate the performance of the learner.

2.3 Exploratory Data Analysis

Exploratory Data Analysis or EDA is mindset or an approach towards the analyzing data, that can be described by four main characteristics:

- Emphasis on understanding data. Trying to answer the broad question of "what is going on?" data.
- Emphasis on graphical representation of the data.
- Emphasis on hypothesis generation from data.
- Skepticism and flexibility, when choosing which methods to apply.

(Behrens, 1997)

Techniques used in EDA is usually a set of graphical and statistical methods. Box plots, histograms, scatter plots etc. are used to explore dataset's characteristics. Data linearity, seasonality, trend, autocorrelation, distribution of the data are among insights sought out while performing EDA.

The principles of EDA were developed by Turkey, John Wilder in the early 1970's.

2.4 Regression

Learning problem of regression consist of using data to predict, as closely as possible, the correct real-valued labels of the points or items considered. (Mohri, Rostamizadeh, & Talwalkar, 2018) Main goal is to predict some dependent variable from an explanatory variable in some form. Explanatory variable can be a numerical value, set of features etc. Unlike in the problem of classification, one is not looking for a definite answer. Since the range of possible answers is infinite, one of the most common methods of evaluating a model's accuracy is using MAE (mean squared error) as the loss function.

General process of solving regression problem, also depicted on Figure 4 is:

1. Gather set of data for required problem.
2. Identify that problem you are solving is indeed a regression, i.e., answer lies in infinite range.
3. Define your question of interest or your goal.
4. Perform EDA (or exploratory data analysis) to summarise data's main characteristics.
5. Perform cleaning of the data, to remove damaged data, outliers, null values, and other errors.
6. Perform feature selection, to select specific features which will have the most predictive power towards your variable in question.
7. Apply selected regression algorithm.
8. Evaluate and interpret results.
9. Repeat steps 4 - 8 until your goal is met. During the next cycle changes can be made to any step, for example not removing some values that you considered redundant first, but rather using them in the model or selecting a different regression algorithm.

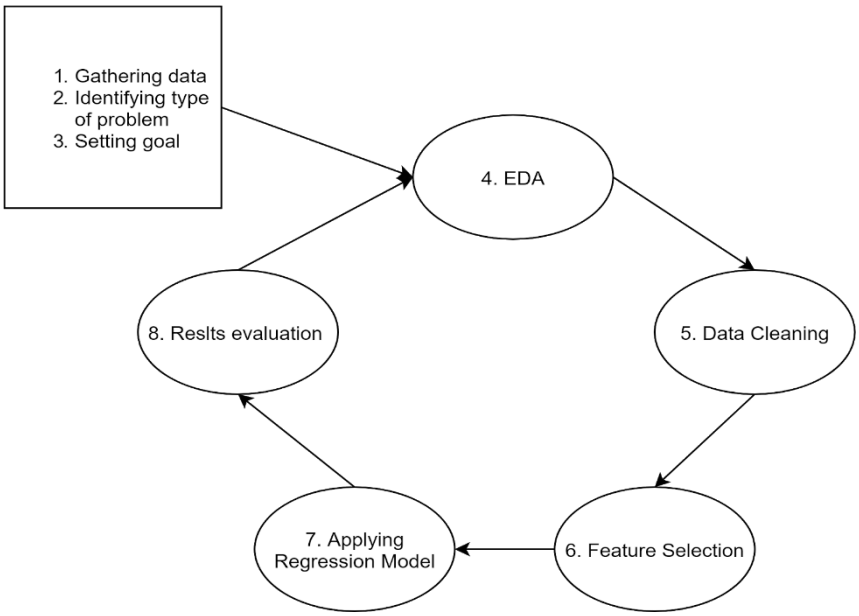

Figure 4. Steps of solving regression problem

The most common way of performing evaluation on step 8 is by using cross-validation. Cross validation will be discussed in detail in Chapter 2.4.4.

2.4.1 Static Models

Static models describe systems contemporaneously, as they exist in the specific time period. Static models can be described as memoryless, in other words one only needs input parameters of the system at the moment, to accurately predict output. (Orkun Öğücü & Saranli, 2012)

Some of the advantages of the static models are:

- They generally require less data to train.
- Require low computational power to build
- Static models generally have simplier, easier to understand structure
- Is required to be build once and then deployed for future used

Some of the disadvantages are:

- Static models cannot adapt to the changes in the data source
- Static models do not perform accurate prediction, if data has strong autocorrelation

They are used for modelling of the system that does not contain autocorrelated data. Data without autocorrelation means that no lagged version of the data is correlated to the current version. So static models are built for the systems that do not have strong autocorellaiton nor seasonality (data with seasonality means, that there is a trend in data repeating on the regular intervals), also data source of the model should not update often, to keep predictive power of the model overtime.

2.4.2 Dynamic Models

Dynamic modeling could be described as having a memory or needing input parameters in previous timestamps to make accurate prediction on an explored variable. In other words data gathered in those systems is autocorrelated and/or has seasonality. Dynamic models could also mean models, that are deployed online and are constantely updated. (West & Harrison, 1997)

Some of the advantages of the dynamic models are:

- Ability to learn data seasonal trends of data.
- Better ability to learn non-linear behavior, compared to static models.

Some of the disadvantages are:

- Usually, more data is required, idealy multiple seasons of data.
- Higher computational complexity.

2.4.3 Linear Regression

Linear regression is the most common regression model. Process of Linear Regression consists of finding such line equation parameters, so empirical mean squared error is smallest. (Mohri, Rostamizadeh, & Talwalkar, 2018) LSE (Least Squared Error) is used for optimization of the line. Linear regression line equation can be expressed as:

$$y(X) = wX + b \quad (1)$$

Where y is the dependant variable, X is explanatory. w is the slope of the line and b is an intercept. Intercept means the value of the y in case of X being 0. Optimization problem for the line can, for range $i_1 \ldots\ldots i_n$ can be expressed as follows:

$$\min_{w,b} \frac{1}{n} \sum_{i=1}^{n} (w(X_i) + b - y_i)^2 \quad (2)$$

Line is fitted such as LSE for all her points is minimal. Example can be seen on Figure 5.

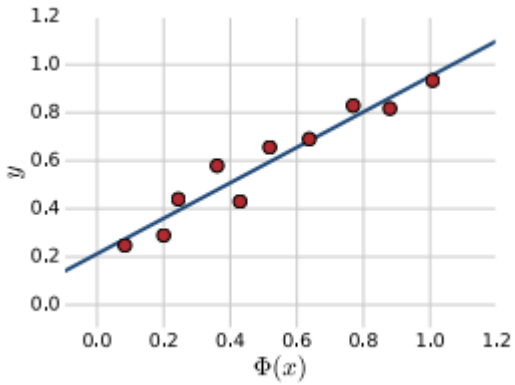

Figure 5. Example of line fitting. (Mohri, Rostamizadeh, & Talwalkar, 2018)

2.5 Metrics and optimization methods

2.5.1 Loss Function

Loss function is used to determine the error (or the “loss”) between the output of an algorithm and given target value. (Loss Function, n.d.)While fitting a machine learning algorithm for supervised learning problems, one always tries to find the minimum of a loss function. There are several popular loss functions for regression problems, most common of them are discussed in this chapter:

- Mean Square Error
- Mean Absolute Error
- Huber Loss
- Log cosh Loss
- Quantile Loss

While plotting loss functions, a situation where validation value is equal to 100 and predicted values range from -10000 to 10000 was considered.

2.5.2 Mean Squared Error

Mean Squared Error is the sum of square distance between predicted and validation variables.

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n} \quad (3)$$

In Formula 3 y_i is the validation variable, $\hat{y}_i$ is the predicted variable and n is the number of predictions.

Mean Squared Error is one of the most used loss functions for regression. On Figure 6, the loss function is plotted.

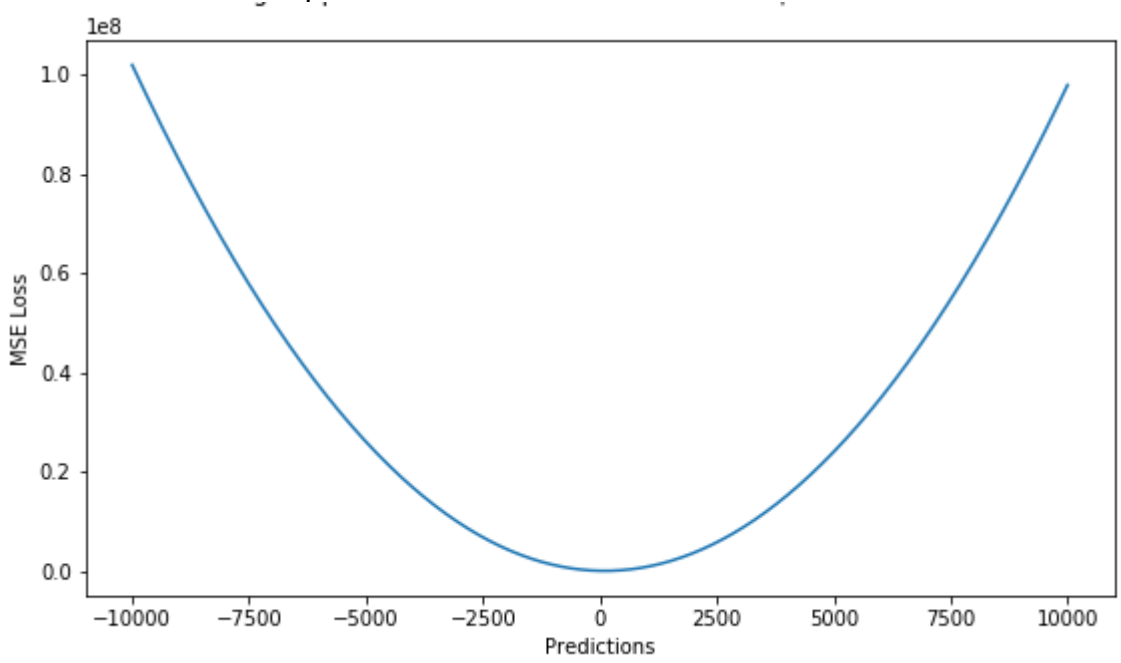

Figure 6. Distribution of loss for MSE. (Prince, 2018)

2.5.3 Mean Absolute Error

Mean Absolute Error is the sum of absolute difference between predicted and validation value.

$$MAE = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n} \quad (4)$$

Formula 4 y_i is the validation variable, $\hat{y}_i$ is the predicted variable and n is the number of predictions.
Mean Absolute error is another popular loss function for regression. Advantage of MAE over MSE is that it is less susceptible to outliers in the data. Disadvantage is that during learning MAE is more vulnerable to overshooting because gradient (slope of the curve) is consistent. Plot can be seen on Figure 7.

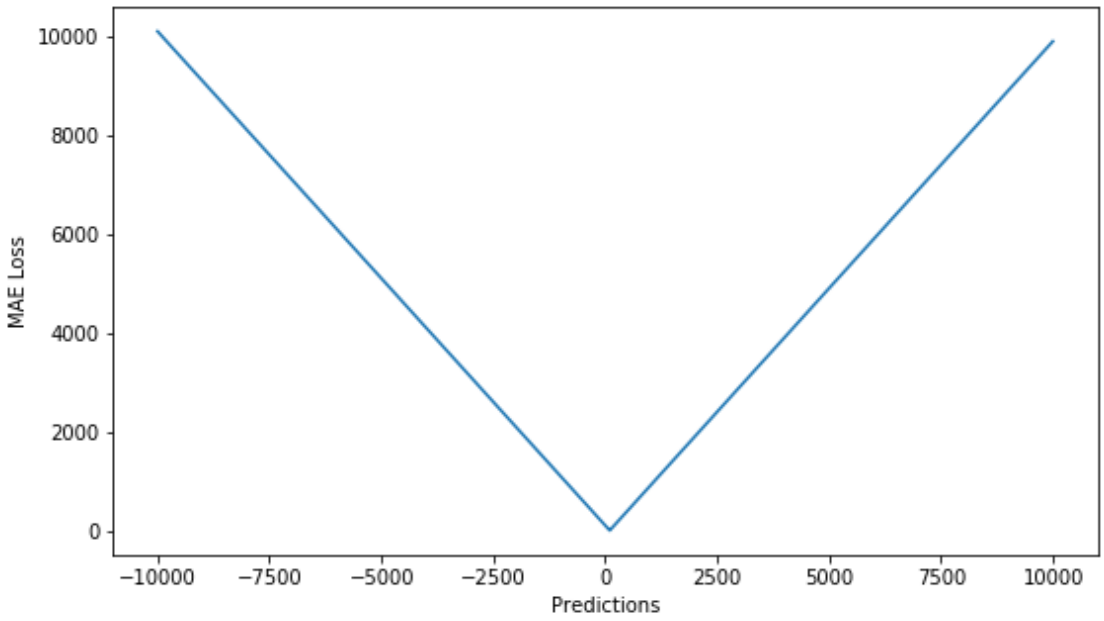

Figure 7. Distribution of loss for MAE. (Prince, 2018)

2.5.4 Cross-validation

Cross-validation is one of the most used data resampling methods to estimate the true prediction error of models and to tune parameters. (Berrar, 2018)

During cross-validation multiple training sets are created with random sampling from the data set. Goal of the cross-validation is to evaluate the intermediate model on the subsets of data not seen during the training. Average performance of the intermediate models is used to estimate performance of the final model. During fitting accuracy on training and testing sets are gathered.

Cross-validation helps to predict true predictive power of the model since data is being tested on multiple subsets of the data.

While performing feature selection, cross-validation helps to identify the optimal number of features. Decreasing number of features helps to avoid overfitting model and decrease computational power required to train the model.

2.5.5 Select K best

Select K Best method helps to choose model parameters for training by assessing importance score between each feature and label. Multiple score functions can be used for obtaining feature score, for example: Mutual Information, ANOVA, regression, family-wise error rate. (sklearn.feature_selection.SelectKBest, n.d.)

Most important parameter is K which determines the number of features selected by their top score.

2.5.6 Repeated KFold Cross-validation

Repeated KFold cross-validation helps to tune model parameters. Data is divided into K subsets. Sampling is done so that no subsets are overlapping. Training is done on K-1 sets and then final evaluation of the accuracy is done on remaining data. During training multiple configurations, such as different numbers of features, are used on different subsets.

2.5.7 Grid Search

The Grid search is the process of searching through explicitly specified subsets of the hyperparameter space of the learning algorithm, search is normally controlled by some performance metric. (Dufour & Neves, 2019) . Sets of hyperparameters is selected before starting Grid Search, for example with Select K Best method, optimal number of features is explored during Grid Search. During this method each set of hyperparameters is evaluated with selected cost function (error of the training on the whole set). This way optimal hyperparameters for model could be selected.

3 MODELLING

3.1 Vuores D-wing automation system

Vuores school was aiming to reduce their energy consumption for heating and air conditioning, to achieve that building automation system was decided to be optimized. Optimization with means of Machine Learning was developed for testing such an approach.

Given the large area of the building and the complexity of its HVAC system, experiments were set to be done, exclusively, in the D-wing of the building.

There were two main heating sources coming to the Vuores D-wing. The first one was the PV03 battery network responsible for heating of air in all the floors. Second one was the IV01 ventilation that feeds the TK01D heat exchanger, which is later used for heating air in classes, corridors, and offices. It is important to note that the IV01 system also feeded the A and C wings.

Three parameters were selected for modelling in the D-wing:

- Energy PV03, being the larger part of energy consumption in the wing.
- Average room temperature, being important metric of Indoor Air Quality
- Average room CO2, also being important Indoor Air Quality metric

IV01 was not selected for modelling since only a partition of its power was used in the D-wing, and it was impossible to determine the exact percentage of this partition.

TK01D did not have a measurement point, which would have allowed us to model its power consumption.

A schema of the D-wing heating supply can be seen in Figure 8.

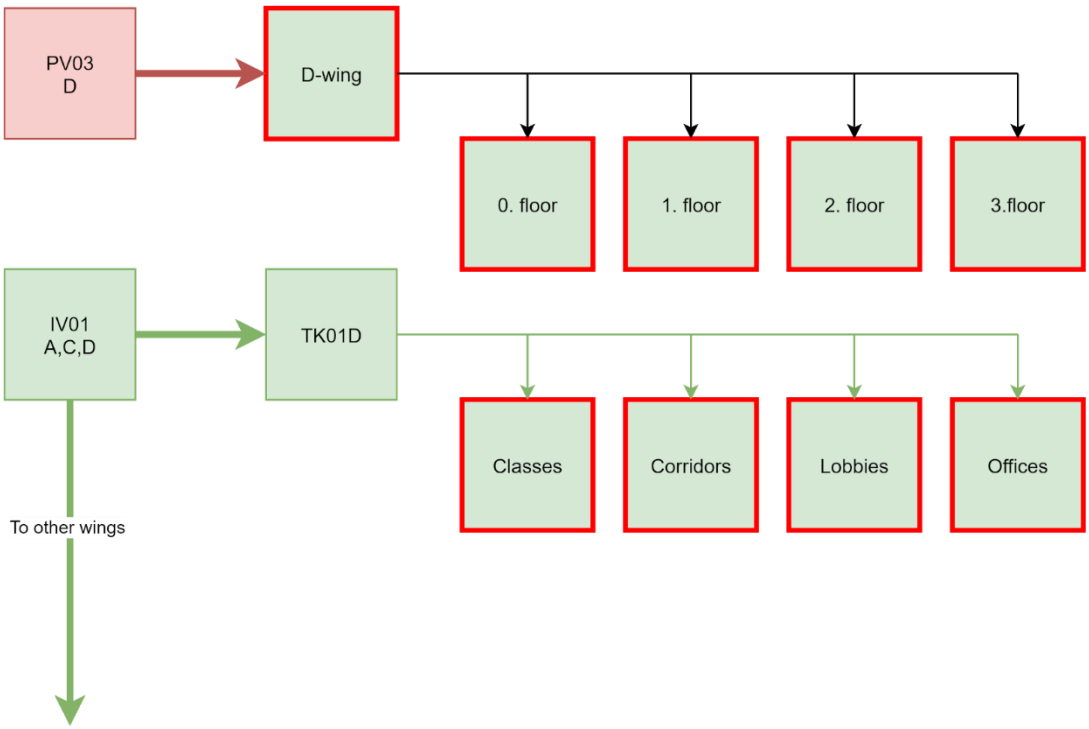

Figure 8. Schema of D-wing heating

3.2 Tools selected for the modelling

Since parameters selected for modelling lie in infinite range, the problem was classified as regression. Final goal of the model was to give accurate predictions on those parameters, while avoiding overfitting of the model.

Python 3.9 programming language was used for its great selection of Machine Learning and statistical analysis libraries. This programming language is also the one author has wide experience with. Full list of installed dependencies can be found in Appendix 2. Key tools that were used:

- Pandas – is a popular for data analysis and data manipulation library.
- Scikit-learn – is a free library that features tools for predictive data analysis.
- Matplotlib – is a plotting library for data visualization in python.

Functions from Scikit-learn library were mainly used for implementing the modelling. Key of those are:

- ‘train_test_split ‘ from Scikit-learn Model Selection was used for dividing data into training and testing sets.

- ‘LinearRegression’ from Scikit-learn Linear Model was used for building linear regression model.
- ‘MinMaxScaler’ from Scikit-learn Preprocessing for 0 to 1 scaling of the data.
- ‘GridSearchCV ‘ from Scikit-learn Model Selection was used for performing Grid Search on the data.
- ‘SelectKBest‘ from Scikit-learn Feature Selection was used for selecting features of the regression model.

Code can be found at appendix 4.

3.3 Data Description and EDA

Data from the Vuores Building was gathered through sensors integrated into BAS (building automation system) and stored into a SQL database. Approximately seven months of data was collected at the time. The database included many parameters available for the analysis, it included measurements of physical quantities as well as BAS’s control parameters, for example a setpoint used to control of indoors temperature.

The data was presented for the author in a CSV (Comma Separated Values) format seen in Table 3. (Note that columns there were renamed from the Finnish language for the author’s convenience)

	item_id	inserted_at	localtime_Helsinki	value	unit	type	Measurement point	Measurement	Key	System ID	System Name	System Number	Device ID	Device Name	System Location(Wing)	Device Location Number	Device Location Name
0	21/Tampere-Vuoreksen koulukeskus-AS4/IO Bus/SI...	2020-04-30 18:42:13	2020-04-30 21:42:13	1	none	boolean	TK01C-TZA-K	TZA	Ylilämpö/jäätymisvaaratermostaatti tila	TK	Tuloilmakojeisto	1.0	TZA_K	NaN	C	NaN	NaN
1	21/Tampere-Vuoreksen koulukeskus-AS4/IO Bus/SI...	2020-04-30 18:42:14	2020-04-30 21:42:14	1	none	boolean	TK01C-PF02-TÄYS-K	PF	Poistoilmapuhallin tila	TK	Tuloilmakojeisto	1.0	PF	Poistoilmapuhallin	C	2.0	Tuloilmakanavassa LTO:n jälkeen
2	21/Tampere-Vuoreksen koulukeskus-AS4/IO Bus/SI...	2020-04-30 18:42:14	2020-04-30 21:42:14	1	none	boolean	TK01C-PF02-OSA-K	PF	Poistoilmapuhallin tila	TK	Tuloilmakojeisto	1.0	PF	Poistoilmapuhallin	C	2.0	Tuloilmakanavassa LTO:n jälkeen
3	21/Tampere-Vuoreksen koulukeskus-AS4/IO Bus/SI...	2020-04-30 18:42:14	2020-04-30 21:42:14	1	none	boolean	TK02C-FG01-K	FG	Pellin toimilaite	TK	Tuloilmakojeisto	2.0	FG	Pellin toimilaite	C	1.0	Ulkosäleikön jälkeen
4	21/Tampere-Vuoreksen koulukeskus-AS4/IO Bus/SI...	2020-04-30 18:42:14	2020-04-30 21:42:14	1	none	boolean	TK01C-FG31-K	FG	Pellin toimilaite	TK	Tuloilmakojeisto	1.0	FG	Pellin toimilaite	C	31.0	Poistoilmakanavassa

Table 3. Initial look at the data.

Time of insertion to the database was in UTC time zone. Each individual measurement location could be identified by its unique measurement point, but in one location multiple values could be measured, so item_id was used for identification of unique measurements. Data in the table was ordered by time of appearing in the database. Immediate problem was that every set of measurements had different sampling times. Difference was coming from sensors only sending measurement record to the database when sensor value has been updated.

It is hard to perform EDA on such unstructured data, but key findings are that all the important data is in numerical format and that data can be easily resampled with forward fill method. Forward fill method puts the latest value in the time series and fills it instead of NaN value.

After processing described in Chapter 4.2. Some more exploring was performed.

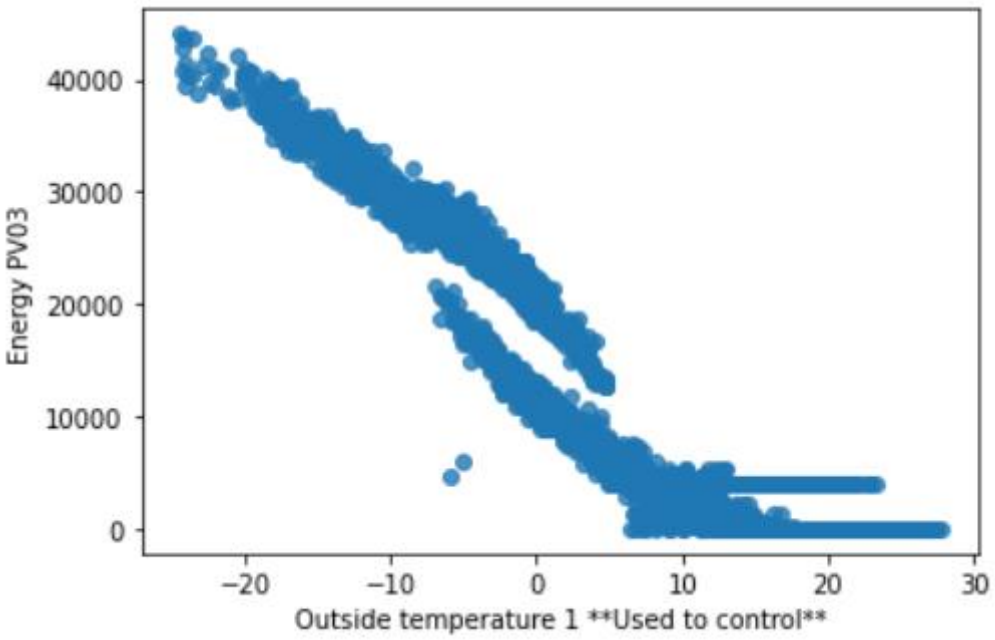

Figure 9. Relation between outdoors air temperature and energy consumption in battery network.

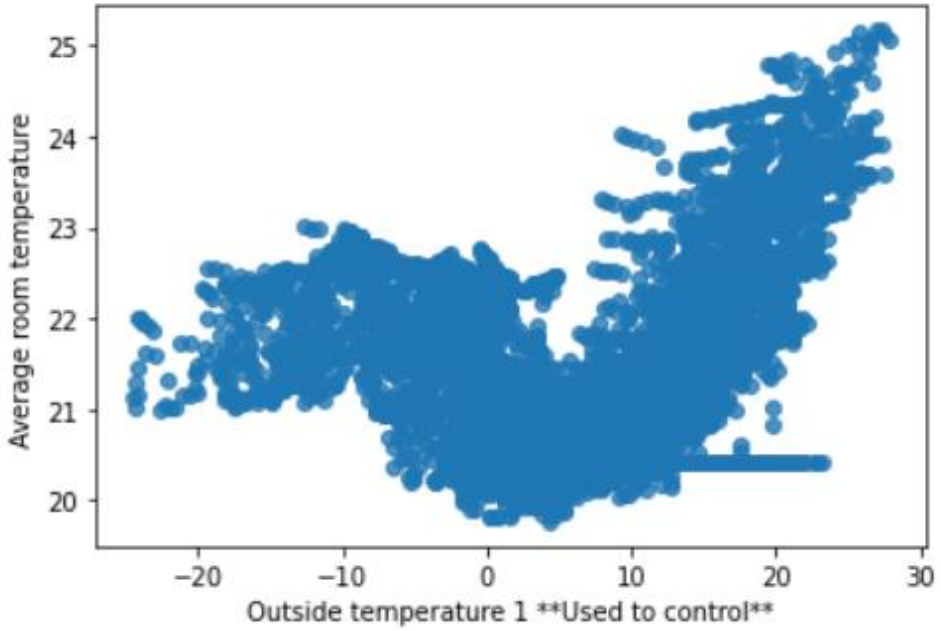

Figure 10. Relation between outdoors air temperature and average indoor air temperature.

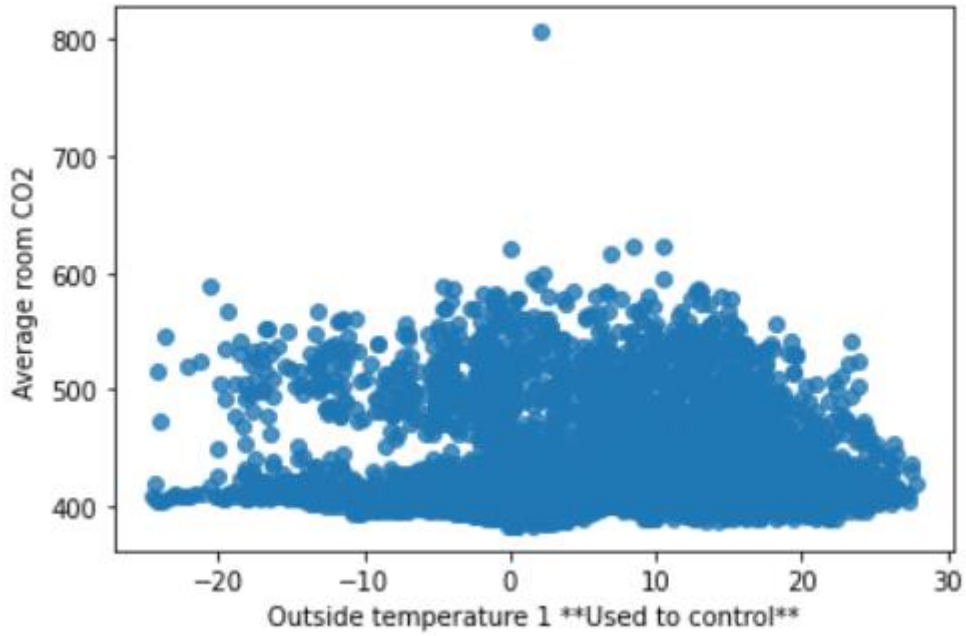

Figure 11. Relation between outdoors air temperature and average indoor air CO2 concentration.

On Figures 9-11 relation of Outside temperature to explored parameters was plotted. Gap in Figure 9 can be explained by different automation system configuration in summer and winter. On Figure 11 can be concluded that CO2 rarely rises above 600ppm level, which well is within S2 level.

3.4 Data Processing

Due to issues described above, the dataset needed heavy modifications before EDA could be performed.

During the first step, a table of all parameters related to the D-wing of the building was created. Table contained pairs of Description (Measurement in form easily understood for humans) and “item_id” (to identify specific measurement). Table 4, full table can be found in appendix 3.

	Description	item_id
2	Energy IV01	11/Tampere-Vuoreksen koulukeskus-AS1/LonWorks Local FT-10 Interface/N_AS01/GR_LON/IV01_QQ01_FQ01/Virtual Functional Block/nvoPowerV1/Value.power_f
3	Energy PV03	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/PV03_QQ01_FQ01/Virtual Functional Block/nvoPowerV1/Value.power_f
4	Flow IV01	11/Tampere-Vuoreksen koulukeskus-AS1/LonWorks Local FT-10 Interface/N_AS01/GR_LON/IV01_QQ01_FQ01/Virtual Functional Block/nvoV1_Flow/Value.flow_f
5	Flow PV03	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/PV03_QQ01_FQ01/Virtual Functional Block/nvoV1_Flow/Value.flow_f
6	TK entering water temperature	11/Tampere-Vuoreksen koulukeskus-AS1/LonWorks Local FT-10 Interface/N_AS01/GR_LON/IV01_QQ01_FQ01/Virtual Functional Block/nvoTemperature1/Value.temp_p
7	TK return water temperature	11/Tampere-Vuoreksen koulukeskus-AS1/LonWorks Local FT-10 Interface/N_AS01/GR_LON/IV01_QQ01_FQ01/Virtual Functional Block/nvoTemperature2/Value.temp_p
8	PV entering water temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/PV03_QQ01_FQ01/Virtual Functional Block/nvoTemperature1/Value.temp_p
9	PV water return temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/PV03_QQ01_FQ01/Virtual Functional Block/nvoTemperature2/Value.temp_p
10	Outside temperature 1 **Used to control**	01/Tampere-Vuoreksen koulukeskus-AS1/IO Bus/Slot08:UI-16/UT01-TE00_M
11	Outside humidity	01/Tampere-Vuoreksen koulukeskus-AS6/IO Bus/15_UI-8.AO-V-4/UT02-ME00_M
12	Outside temperature 2	01/Tampere-Vuoreksen koulukeskus-AS6/IO Bus/15_UI-8.AO-V-4/UT02-TE00_M
13	Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.001/TKXXTC20XX/TC20/TE20_M
14	Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.002/TKXXTC20XX/TC20/TE20_M

Table 4. Measurements taken in the D-wing.

Because of the method that sensors record data in the Vuores building's automation system, practically any sample rate could have been selected. One could always look on the previous recorded value to select current at any sampling interval. One hour sample rate was selected for the purpose of this project, with trial and error it was discovered that at this rate there is enough data to build accurate models and run cross-validation without increasing computational complexity with higher frequency of the data. Data was resampled with forward fill method. Measurements of the CO2 and air temperature were recorded across multiple sensors in multiple classrooms. Average values of those measurements at time of recording were selected to represent air temperature and CO2 concentration of the system. As the result of data modifications, we have a dataset, where each measurement is as an individual column indexed by time.

inserted_at	Energy IV01	Energy PV03	Flow IV01	Flow PV03	TK entering water temperature	TK return water temperature	PV entering water temperature	PV water return temperature	Outside temperature 1 **Used to control**	Outside humidity	...	TK air humidity	Air entering temperature	Air leaving Pressure	TK return water temperature	PF speed %	TF speed %	TK heating valve position	Air leaving CO2	Average room temperature	Average room CO2
2020-05-03 10:00:00	14200.0	3700.0	2.16667	0.425000	29.10	27.54	26.27	24.09	10.08	64.16	...	75.73	19.71	20.85	22.16	60.14	60.11	52.325	419.6	21.099496	402.512089
2020-05-03 11:00:00	13100.0	2900.0	2.13333	0.425000	28.70	27.25	25.43	23.80	10.72	54.96	...	74.38	19.82	20.88	21.90	60.14	60.11	52.325	420.8	21.109399	401.863192
2020-05-03 12:00:00	12800.0	2600.0	2.11667	0.425000	28.50	26.93	25.03	23.54	10.99	60.32	...	73.40	19.62	20.85	21.62	60.14	60.11	51.250	419.6	21.132677	401.041002
2020-05-03 13:00:00	13900.0	4000.0	2.10000	0.426944	28.91	27.30	25.98	23.94	10.66	63.75	...	72.89	19.74	20.88	22.05	60.14	60.11	51.250	429.2	21.133561	401.057182
2020-05-03 14:00:00	11700.0	2600.0	2.10000	0.425000	28.38	27.03	25.10	23.64	11.47	56.91	...	72.81	19.59	20.88	21.65	60.14	60.11	51.250	419.6	21.134938	401.307800

Table 5. Data after processing.

3.5 Feature Selection

First step in feature selection will be determining the importance score of different parameters. To do so, data was scaled 0 to 1 for all parameters, as seen on example in Table 6.

	Energy IV01
0	0.098270
1	0.090657
2	0.088581
3	0.096194
4	0.080969

Table 6. Example of scaled parameter.

Feature score was calculated with SelectKBest method. Results can be seen on picture 12. Feature with the highest score for the “Energy PV03” was “PV entering water temperature”, which is the temperature of the water entering the battery network.

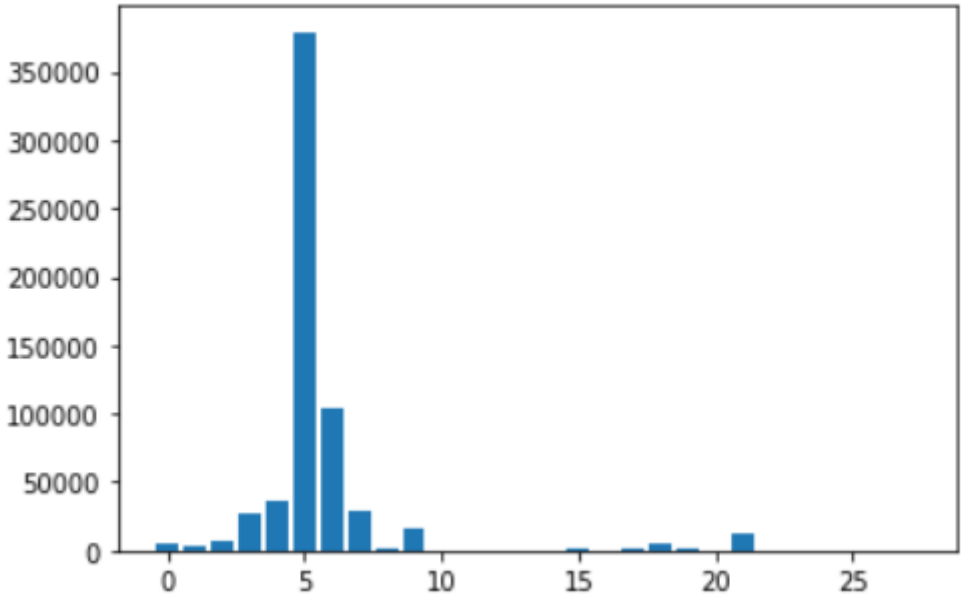

Figure 12. Feature score for energy

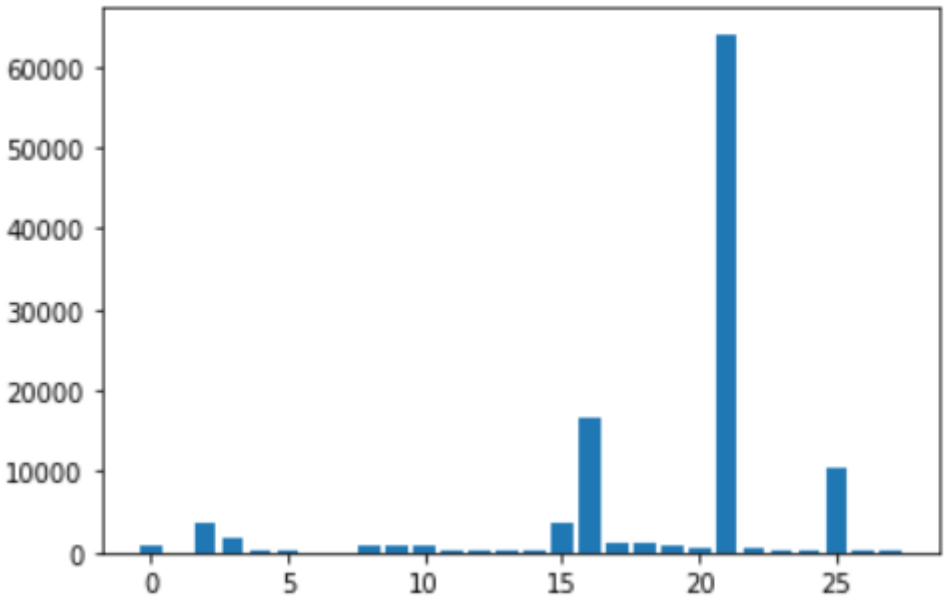

Figure 13. Feature score for temperature

For the “Average room temperature” “Air leaving Pressure .1” had the highest score. Can be seen on the Figure 13.

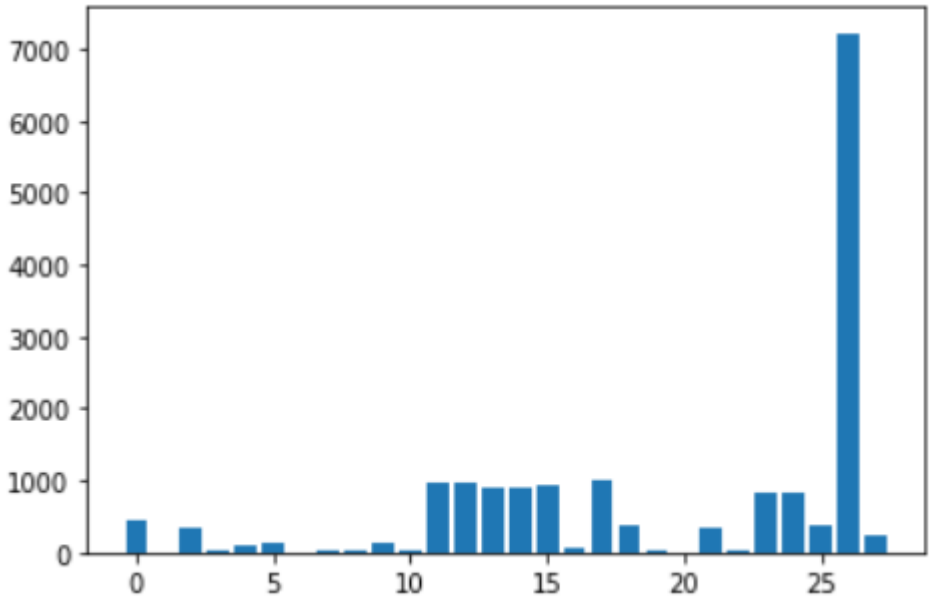

Figure 14. Feature score for CO2

For the “Average room CO2” “Air leaving CO2”, which is measurement of CO2 concentration returning to the ventilation system, had the highest feature score.

During the feature selection number of the features to use in the model will be determined with Grid Search method. Features will be sorted in highest to lowest feature score, then K best of them will be selected for modelling. Number of K will be optimized during Grid Search.

3.6 Modelling

Based on the feature assessment performed in the previous chapter. Modelling was performed using Repeated K-Fold cross-validation with Grid Search method. This method was selected because of its known ability to optimally predict number of features to use in the model. (Bao & Liu, 2006) Linear regression model was built with features selected during grid search.

First the baseline solution with all available features was created to compare to accuracy of the models fitted with feature selection.
Process of modelling of each parameter can be seen in figure 15.

Steps of modelling each value are:

- Create repeated K fold evaluation method. In this step number of splits (subsets of data to perform training on) and number of repeats is defined. Random seed can be explicitly specified.
- Define model to evaluate. During this step evaluation method was selected. With Select K Best method. K best features will be selected. For example, with K being equal to 5, 5 top scoring features will be used as hyperparameters for a model.
- Define grid and grid search. In this grid search parameters are defined. Most important is to select loss function. At this project Negative MSE was used.

MSE was negative because Sci-kit learn GridSearchCV function works by maximizing this parameter.

- Perform grid search. After performing this step highest value of Negative MSE and set of hyperparameters used to get this error value are returned to the user.
- Select model with best accuracy. At this step best estimator selected at previous step is chosen for future validation.

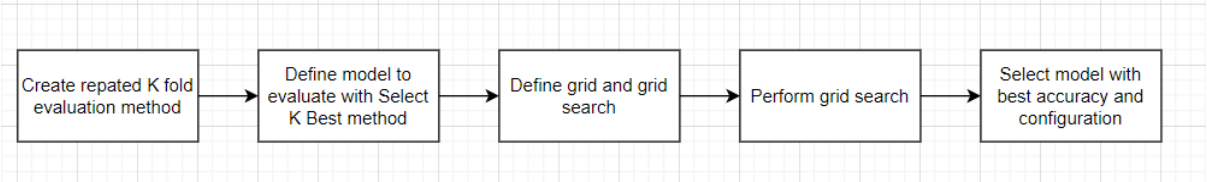

Figure 15. Modelling steps.

All models with optimized features were able to beat the baseline solution.

3.6.1 Parameters selected for modelling

- Energy PV03 model

  For the energy lowest training scoring is achieved with 16 best parameters. Those parameters are:

  'TK return water temperature.1', 'TK return water temperature alt id', 'TK return water temperature', 'TK heating valve position.1', 'TK heating valve position', 'TK entering water temperature', 'TK air humidity', 'TF speed %', 'PV water return temperature', 'PV entering water temperature', 'PF speed %', 'Outside temperature 2', 'Outside temperature 1 **Used to control**', 'Outside humidity', 'Flow PV03', 'Flow IV01'.

- Average room temperature model

  For the temperature lowest training scoring is achieved with 17 best parameters. Those parameters are:

  'TK return water temperature.1', 'TK return water temperature alt id', 'TK return water temperature', 'TK heating valve position.1', 'TK heating valve position', 'TK entering water temperature', 'TK air humidity', 'TF speed %', 'PV water return temperature', 'PV entering water temperature', 'PF speed %', 'Outside temperature 2', 'Outside temperature 1 **Used to control**', 'Outside humidity', 'Flow PV03', 'Flow IV01', 'Fan speed output'.

- Average room CO2 concentration model

For the CO2 concentration lowest training scoring is achieved with 15 best parameters. Those parameters are:

'TK return water temperature.1', 'TK return water temperature alt id', 'TK return water temperature', 'TK heating valve position.1', 'TK heating valve position', 'TK entering water temperature', 'TK air humidity', 'TF speed %', 'PV water return temperature', 'PV entering water temperature', 'PF speed %', 'Outside temperature 2', 'Outside temperature 1 **Used to control**', 'Outside humidity', 'Flow PV03'.

3.7 Validation

3.7.1 Validation of the tuned model

For validation an independent data set containing two days of data was used. Results of prediction can be seen on Figures 16-18.

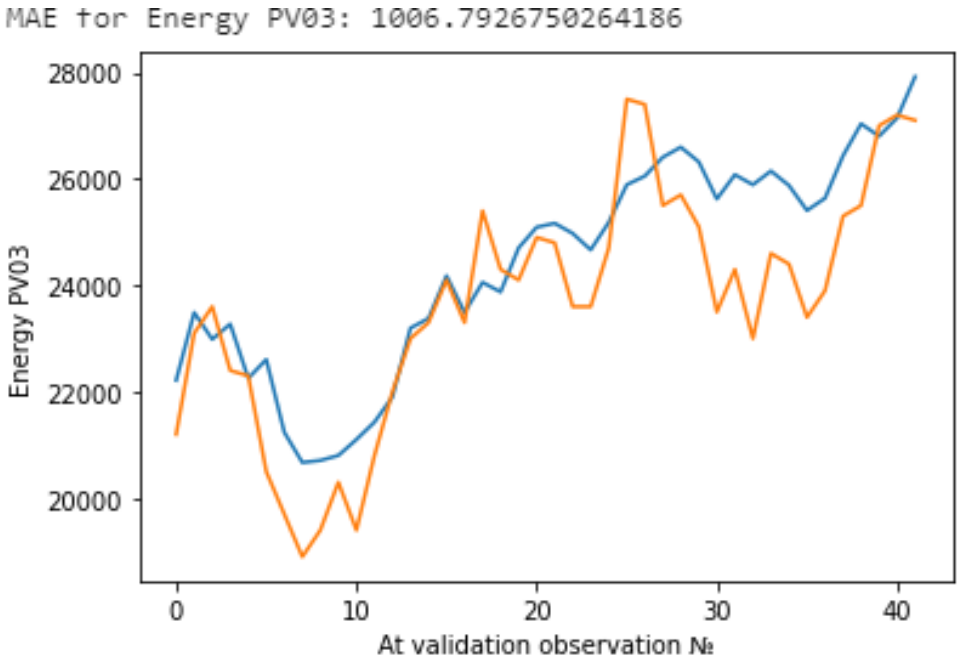

Figure 16. Validation on Energy

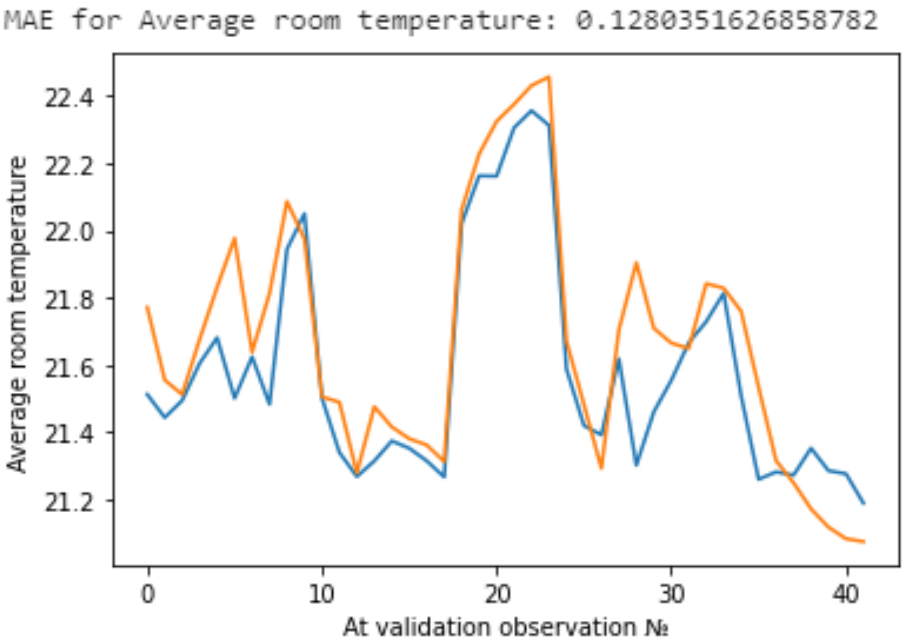

Figure 17. Validation on temperature

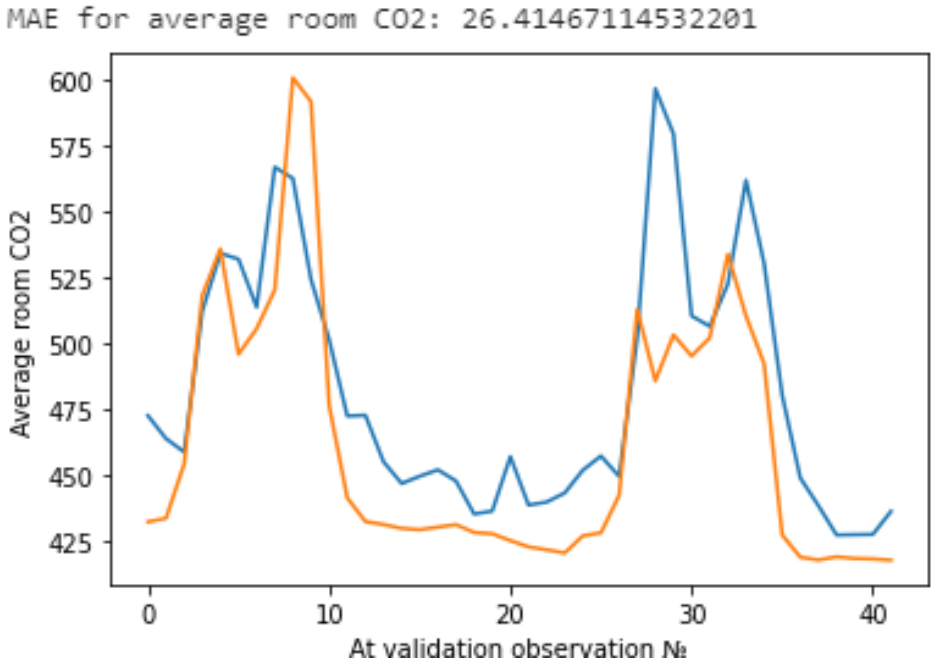

Figure 18. Validation on CO2

Based on the results of predicting on the validation set, it can be concluded that the model learned to generalize and predict on the unseen data of the similar structure.

3.7.2 Comparison to a baseline model

On the table 4 comparison to a baseline (with all features included) models can be seen. Mean Absolute Error value is shown in each cell. In this example Random

number of K is equal to 5, here it is an arbitrary number lower that optimal number of K.

	Baseline (all features included)	Optimal K of features	Random number of K
Energy Model	1054 W	1006W	1097W
Temperature model	0.135 C°	0.128 C°	0.131 C°
CO2 model	27.5 ppm	26.4 ppm	28.1 ppm

Table 7 Comparison of optimal and baseline models.

As can be seen in the ‘Random number of K’ columns using some lower than optimal K number of parameters, results are inconsistent compared to the baseline model. The temperature model was able to improve, whereas the Energy and the CO2 models are performing worse that their baseline. The model with optimal K of features is best performing in all 3 examples.

4 POSSIBLE APPLICATIONS

A possible application of the models in the project would include using them in the building automation control systems for forecasting.

- Energy model can be used for an algorithmic optimization like Particle Swarm Optimization or Genetic Algorithm. It should help to decrease energy consumption over time. Optimization of HVAC systems in discussed in detail in Chapter 6.1.
- Air temperature and CO2 can be used for predicting if the HVAC system will stay within air quality standards if parameters of ventilation of heating to be changed. This will ensure that adjustment of the parameters to save energy, will not affect comfort of the humans inside the building.

- Detecting anomalies or faults. If some anomaly occurs, for example due to the equipment malfunctioning, then this would result in a great difference between predicted and observed value.

4.1 Application to modern HVAC optimization

A modern approach to optimizing HAVC systems consists of selecting optimal setpoints by suing a model to predict load of the system, define setpoints and then check how these setpoints would affect conditions of the building. (Nassif, 2005) A diagram of the process can be seen in Figure 19. Key components of the optimization system for HVAC systems are:

- Interface for processing gathered data. Building data could be gathered by a set of sensors external to the HVAC system. In this case interface is required to change data to the format understood by HVAC controls and combined with HVAC system's internal sensors. Also, if prediction tool is not native to the HVAC, some processing of data could be required.
- Prediction tool. A model or a combination of models capable of predicting load on the HVAC system in curtain conditions.
- Model for selecting setpoints. An algorithmic solution, for example based on genetic algorithm, for selecting optimal setpoint for the system. Optimal here refers to a setting of the system that will maintain good Indoor Air Quality with minimum energy consumed.
- Model for testing setpoints. A model or combination of models for testing system with updated setpoint. This model always communicates with model for selecting setpoints.
- Optimal solution selection tool. A tool that selects most optimal set of settings and applies them to system's control.

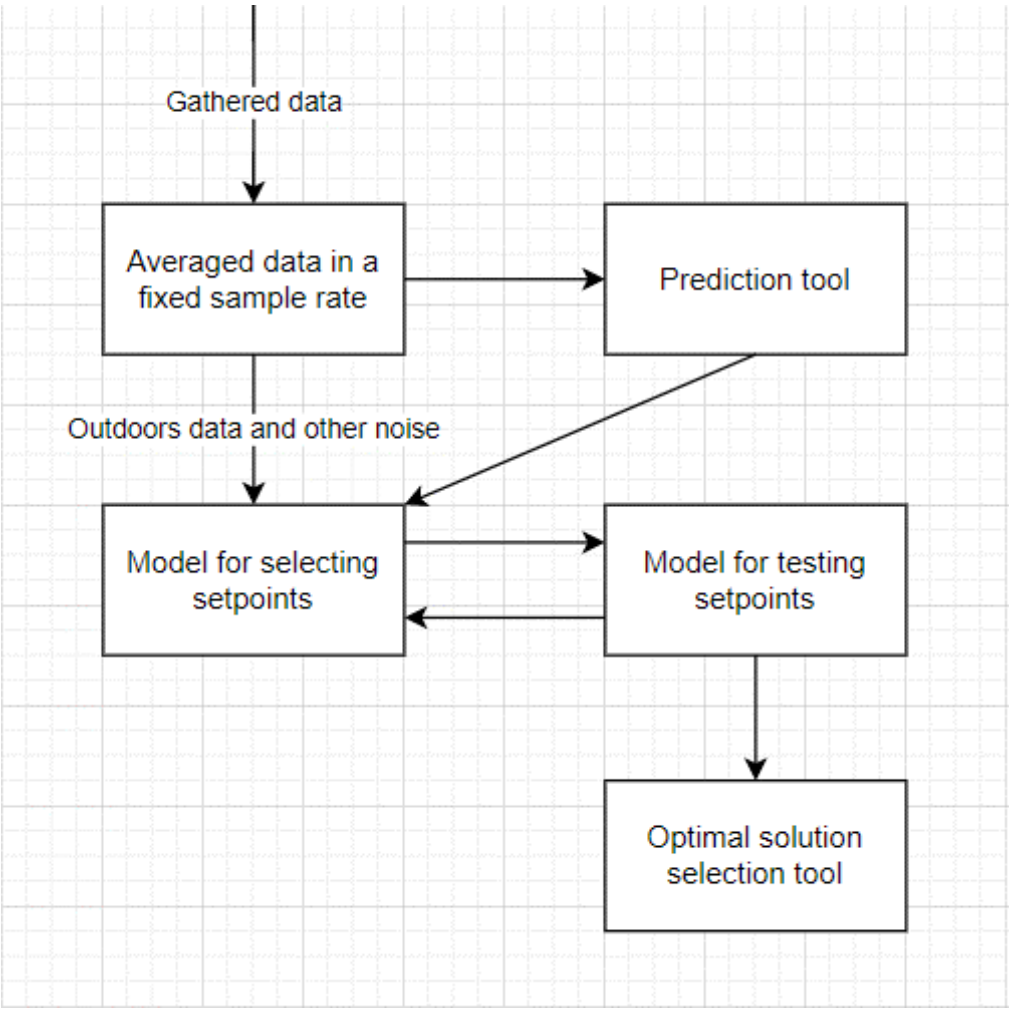

Figure 19. Optimization process of the a HVAC system.

The model developed for predicting battery network heating energy consumption could be used as the prediction tool in optimization process. Predicted load will be used in Model for selecting setpoints for the system. The effect on the Indoors Air Quality of the HVAC with suggested setpoints could be tested with developed average rooms temperature and average rooms CO2 models.

Advantage of using a machine learning model as the prediction tool is that such approach will allow to learn overtime individual behavior of the building. Each building set has unique feature, that are affecting energy consumption of the HVAC system and are hard to specify for the system with conventional models. Machine Learning algorithm can learn the effect of those unique features, without exactly knowing them.

Such approach could save up to 12,4% energy in ideal conditions, according to the experiment conducted by The University of Iowa. (Wei, Kusiak, Li, Tang, & Zeng, 2015). It is important to note, that the mentioned experiment was performed on the small scale, in the building consisting of 3 middle sized rooms and with only 2.5 weeks of summertime data. It could be expected that on a large scale it will be almost impossible to reach such high savings.

5 CONCLUSION

5.1 Possible improvements

Main limitations of the model appeared because initial data has very little variance, which is a very common issue in building automation. Since the model has limited extrapolation capabilities it prevents from making accurate predictions on the data with unusual configuration of building automation system. Possible solution the problem is:

- Perform special situations, where parameters of the HVAC system will be adjusted for testing while gathering data. They should be run in time when the building is not used for some period, to not disturb building's operation.
- Gathering more data will help any model. At least one year of data should be collected to make accurate predictions in different seasons.
- Including data from whole building, not only D-wing. It will allow to better learn building's behavior and to better utilize parameters, which were considered a noise for a D-wing, for example ventilation energy consumption IV01.

5.2 Conclusion

In this project, a machine learning regression model was built for simulating behavior of the Vuores building automation system. Energy consumption in the battery network, air temperature and air CO2 concentration were modeled using a regression model with optimized feature selection. The final energy model was able to predict the energy consumption with MAE of 952 on the validation set. MAE for temperature and CO2 concentration models were 0.13 and 27.4 respectively. Main application for those models could be found in the optimization of the HVAC systems.

References

Bao, Y.;& Liu, Z. (2006). *A Fast Grid Search Method in Support Vector Regression Forecasting Time Series.* Springer.

Behrens, J. T. (1997). *Principles and Procedures of Exploratory Data Analysis.* Arizona. Noudettu osoitteesta https://core.ac.uk/download/pdf/193648223.pdf

Berrar, D. (January 2018). *Cross-validation.* Noudettu osoitteesta ResearchGate: https://www.researchgate.net/publication/324701535_Cross-Validation#pf7

Dufour, J.-M.;& Neves, J. (2019). Conceptual Econometrics Using R. Teoksessa *Handbook of Statistics.* Noudettu osoitteesta https://www.sciencedirect.com/topics/mathematics/grid-search#:~:text=Grid%20search%20is%20a%20process,independently%20using%20a%20probability%20distribution.

Ian, K. (2012). Assessing electrical energy use in HVAC systems. *Rehva*, 6-12.

Loss Function. (ei pvm). Noudettu osoitteesta Deep AI: https://deepai.org/machine-learning-glossary-and-terms/loss-function

mindmingles. (3. 12 2020). *5 Aspects to Consider When Buying a New HVAC System*. Noudettu osoitteesta AndroClue: https://androclue.com/when-buying-a-new-hvac-system/

Mohri, M.;Rostamizadeh, A.;& Talwalkar, A. (2018). *Foundations of Machine Learning* (2nd p.). The MIT Press. Noudettu osoitteesta https://cs.nyu.edu/~mohri/mlbook/

Nassif, N. (2005). *OPTIMIZATION OF HVAC CONTROL SYSTEM STRATEGY USING TWOOBJECTIVE GENETIC ALGORITHM.* Thesis, MONTRÉAL. Noudettu osoitteesta https://espace.etsmtl.ca/id/eprint/340/1/NASSIF_Nabil.pdf

Orkun Öğücü, M.;& Saranli, A. (October 2012). *A Comparative Study on Modeling a Nonlinear Static System by Using Identification Techniques.* Noudettu osoitteesta ResearchGate: https://www.researchgate.net/publication/258221883_A_Comparative_Study_on_Modeling_a_Nonlinear_Static_System_by_Using_Identification_Techniques

Prince, G. (5. June 2018). *5 Regression Loss Functions All Machine Learners Should Know.* Noudettu osoitteesta Heart Beat: https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0

RAY, B. (2015). *What's So Cool About Smart HVAC Systems?* LINK LABD+S. Noudettu osoitteesta https://www.link-labs.com/blog/smart-hvac

Säteri, J. (2002). FINNISH CLASSIFICATION OF INDOOR CLIMATE 2000: REVISED TARGET VALUES. *IRBNET*. Noudettu osoitteesta https://www.irbnet.de/daten/iconda/CIB7264.pdf

sklearn.feature_selection.SelectKBest. (ei pvm). Noudettu osoitteesta Ski-learn: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html

Sugarman, S. (2005). *HVAC Fundamentals.* THE FAIRMONT PRESS, INC. Noudettu osoitteesta https://www.engineeringbookspdf.com/mechanical-engineering/hvac/

Vuoreksen koulu. (20. 8 2020). Noudettu osoitteesta Tampere: https://www.tampere.fi/varhaiskasvatus-ja-koulutus/esiopetus-ja-perusopetus/koulut/vuoreksen-koulu.html

Wei, X.;Kusiak, A.;Li, M.;Tang, F.;& Zeng, Y. (2015). *Multi-objective optimization of the HVAC (heating, ventilation, and air conditioning) system performance.* Iowa.

West, M.;& Harrison, J. (1997). *Bayesian Forecasting and Dynamic Models* (2nd p.). New York: Springer. Noudettu osoitteesta http://213.230.96.51:8090/files/ebooks/Iqtisodiyot/West%20M.,%20Harrison%20J.%20Bayesian%20forecasting%20and%20dynamic%20models%20(Springer,%201997)(ISBN%200387947256)(695s)%20GL%20.pdf

List of Figures

List of tables

Appendix 1: Notes on Indoors Air Quality

For table 1

* In Category S1, the room temperature shall be adjustable in each room/apartment between 20- 24°C. If several people occupy the same room, the target level of the room temperature shall be 21-22°C in the winter and 23-24°C in the summer.

** The set value of the room temperature shall be in the range mentioned in item "Room temperature".

*** The room temperature shall never exceed +35°C. When the outdoor temperature is below +15°C, the room temperature shall not exceed +27°C.

I. The room temperature is usually the air temperature in the occupied zone. The target values apply to conditions where the individual control of room temperature is not utilized. In Category S1, the room temperature may temporarily deviate from the target values for a maximum of three days in the summer and three days in the winter, in design weather conditions. In Category S2, the room temperature may temporarily deviate from the target range for a maximum of seven days in the winter and seven days in the summer in design weather conditions.

II. The air velocity is the omnidirectional 3-minute average air velocity in the occupied zone. It shall be measured in accordance with the Finnish Standard SFS 5511. The measured result and the set target value are presented with a precision of two decimals, the last significant figure being either 0 or 5.

For table 2

I The radon concentration in the room air for new residences determined by the Ministry of Social Affairs and Health shall not exceed 200 Bq/m3 . The annual average concentration of radon in residences shall not exceed 400 Bq/m³. The annual average radon concentration in work places during working hours shall not exceed 400 Bq/m³.

II The concentration of carbon dioxide includes carbon dioxide from outdoor (350 ppm) and human sources. The CO2 concentration can be measured, for example, with an infrared analyzer.

III The concentrations of ammonia and amines in the room air can be measured with an ion-selective electrode or with a photometer). Only the emissions originating from the building materials shall be taken into account, not the emissions from human sources or human activities.

IV The concentration of formaldehyde in the room air can be measured, for example, with a liquid chromatograph (DNPH method) or a chromotrope-acid method in accordance with the Finnish Standard SFS 3862. Only the emissions originating from the building materials shall be taken into account, not the emissions from human sources or human activities.

V The total concentration of volatile organic compounds (TVOC) in the room air shall be measured according to the references (Tirkkonen et al. 1995, Clausen et al. 1993, SFS 5412). At least 70% of the volatile organic compounds shall be identified and the concentration of these compounds shall not exceed known limit values (carcinogens, allergens, the values given in the Indoor Air Instructions, TLV values, etc.). Only the emissions from the building materials shall be taken into account, not the emissions from human sources or human activities.

VI The concentration of carbon monoxide in the room air can be measured with an electrochemical cell or infrared analyzer in accordance with the Finnish Standard SFS 5412.

VII The concentration of ozone in the room air can be measured, for example, with a chemiluminescence or UV absorption method.

VIII The odor intensity of the room air shall be determined with a trained odor panel (ECA 1999).

IX The Classification does not give maximum values for microbe concentrations in room air because there may be mould growth or rot in the constructions of the building even though the microbe concentrations in the air are relatively low. In addition to this, the

microbe concentrations in room air fluctuate greatly depending on time, place, and conditions in the building, as well as on the species of microbe. If the microbe concentration indoors is higher than outdoors and if the species differs from that detected outdoors, this may indicate mould growth in the constructions. Proceedings: Indoor Air 2002 647

X No environmental tobacco smoke odor shall be allowed in spaces where smoking is not permitted. Nicotine can be used as an indicator of tobacco smoke. Nicotine is drawn, for example, into a Tenax tube with the aid of a pump and analyzed by means of gas chromatography. There is tobacco smoke in the room air when the nicotine concentration exceeds 0,05 µg/m³.

XI The PM10 fraction is the mass concentration of airborne particulate matter with an aerodynamic diameter smaller than 10 µm. The mass concentration of airborne particulate matter shall be measured during a 24-hour period in accordance with the Finnish Standard SFS-EN 12341 during normal human activities in the building. If the desired average concentration per day for the selected category is reached indoors, no measurements need to be conducted outdoors. If it is exceeded, the mass concentration of airborne particulate matter shall be measured as simultaneously as possible both indoors and outdoors after which the indoor/outdoor relationship shall be determined. In categories S1 and S2 this relationship may not exceed 0.5; however, the mass concentration of airborne particulate matter indoors shall never exceed 50 µg/m³. For example, if the PM10 concentration indoors is 40 µg/m³, the outdoor concentration of airborne particulate matter shall be measured as well. If the PM10 concentration outdoors is at least 80 µg/m³, the indoor/outdoor relationship is under 0.5, which indicates that the space is acceptable for category S1 with respect to the PM10 concentration.

Appendix 2: List of packages used

‘*package name’==version*

argon2-cffi==20.1.0

async-generator==1.10

attrs==20.3.0

backcall==0.2.0

bleach==3.3.0

cffi==1.14.5

colorama==0.4.4

cycler==0.10.0

decorator==5.0.5

defusedxml==0.7.1

entrypoints==0.3

et-xmlfile==1.1.0

ipykernel==5.5.3

ipython==7.22.0

ipython-genutils==0.2.0

ipywidgets==7.6.3

jedi==0.18.0

Jinja2==2.11.3

joblib==1.0.1

jsonschema==3.2.0

jupyter==1.0.0

jupyter-client==6.1.13

jupyter-console==6.4.0

jupyter-core==4.7.1

jupyterlab-pygments==0.1.2

jupyterlab-widgets==1.0.0

kiwisolver==1.3.1

MarkupSafe==1.1.1

matplotlib==3.4.1

mistune==0.8.4

nbclient==0.5.3

nbconvert==6.0.7

nbformat==5.1.3

nest-asyncio==1.5.1

notebook==6.3.0

numpy==1.20.2

openpyxl==3.0.7

packaging==20.9

pandas==1.2.3

pandocfilters==1.4.3

parso==0.8.2

patsy==0.5.1

pickleshare==0.7.5

Pillow==8.2.0

prometheus-client==0.10.0

prompt-toolkit==3.0.18

pycparser==2.20

Pygments==2.8.1

pyparsing==2.4.7

pyrsistent==0.17.3

python-dateutil==2.8.1

pytz==2021.1

pywin32==300

pywinpty==0.5.7

pyzmq==22.0.3

qtconsole==5.0.3

QtPy==1.9.0

scikit-learn==0.24.2

scipy==1.6.2

seaborn==0.11.1

Send2Trash==1.5.0

six==1.15.0

sklearn==0.0

statsmodels==0.12.2

terminado==0.9.4

testpath==0.4.4

threadpoolctl==2.1.0

tornado==6.1

traitlets==5.0.5

wcwidth==0.2.5

webencodings==0.5.1

widgetsnbextension==3.5.1

xlrd==2.0.1

Appendix 3: Parameters of D-wing

Description	item_id
Energy IV01	11/Tampere-Vuoreksen koulukeskus-AS1/LonWorks Local FT-10 Interface/N_AS01/GR_LON/IV01_QQ01_FQ01/Virtual Functional Block/nvoPowerV1/Value.power_f
Energy PV03	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/PV03_QQ01_FQ01/Virtual Functional Block/nvoPowerV1/Value.power_f
Flow IV01	11/Tampere-Vuoreksen koulukeskus-AS1/LonWorks Local FT-10 Interface/N_AS01/GR_LON/IV01_QQ01_FQ01/Virtual Functional Block/nvoV1_Flow/Value.flow_f
Flow PV03	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/PV03_QQ01_FQ01/Virtual Functional Block/nvoV1_Flow/Value.flow_f
TK entering water temperature	11/Tampere-Vuoreksen koulukeskus-AS1/LonWorks Local FT-10 Interface/N_AS01/GR_LON/IV01_QQ01_FQ01/Virtual Functional Block/nvoTemperature1/Value.temp_p
TK return water temperature	11/Tampere-Vuoreksen koulukeskus-AS1/LonWorks Local FT-10 Interface/N_AS01/GR_LON/IV01_QQ01_FQ01/Virtual Functional Block/nvoTemperature2/Value.temp_p
PV entering water temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/PV03_QQ01_FQ01/Virtual Functional Block/nvoTemperature1/Value.temp_p
PV water return temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/PV03_QQ01_FQ01/Virtual Functional Block/nvoTemperature2/Value.temp_p
Outside temperature 1 **Used to control**	01/Tampere-Vuoreksen koulukeskus-AS1/IO Bus/Slot08:UI-16/UT01-TE00_M
Outside humidity	01/Tampere-Vuoreksen koulukeskus-AS6/IO Bus/15_UI-8.AO-V-4/UT02-ME00_M
Outside temperature 2	01/Tampere-Vuoreksen koulukeskus-AS6/IO Bus/15_UI-8.AO-V-4/UT02-TE00_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.001/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.002/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.003/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.004/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.005/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.006/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.007/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.101/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.102/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.103/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.104/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.105/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.201/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.202/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.203/TKXXTC20XX/TC20/TE20_M
Room temperature	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.204/TKXXTC20XX/TC20/TE20_M
Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.001/TKXXTC20XX/TC20/QE20_M

29	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.001/TKXXTC20XX/TC20/QE20_M
30	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.002/TKXXTC20XX/TC20/QE20_M
31	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.003/TKXXTC20XX/TC20/QE20_M
32	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.004/TKXXTC20XX/TC20/QE20_M
33	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.005/TKXXTC20XX/TC20/QE20_M
34	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.006/TKXXTC20XX/TC20/QE20_M
35	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.007/TKXXTC20XX/TC20/QE20_M
36	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.101/TKXXTC20XX/TC20/QE20_M
37	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.102/TKXXTC20XX/TC20/QE20_M
38	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.103/TKXXTC20XX/TC20/QE20_M
39	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.104/TKXXTC20XX/TC20/QE20_M
40	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.105/TKXXTC20XX/TC20/QE20_M
41	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.201/TKXXTC20XX/TC20/QE20_M
42	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.202/TKXXTC20XX/TC20/QE20_M
43	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.203/TKXXTC20XX/TC20/QE20_M
44	Room CO2	11/Tampere-Vuoreksen koulukeskus-AS5/LonWorks Local FT-10 Interface/N_AS05/TK01DTC20.204/TKXXTC20XX/TC20/QE20_M
45	Fan speed input	01/Tampere-Vuoreksen koulukeskus-AS5/Modbus Master Network/TK01D_PF01/Kierrosnopeus
46	Fan speed output	01/Tampere-Vuoreksen koulukeskus-AS5/Modbus Master Network/TK01D_TF01/Kierrosnopeus
47	Air entering Pressure	01/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot07:UI-16/TK01D-PE10_M
48	Air leaving Pressure	01/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot07:UI-16/TK01D-PE30_M
49	Air entering temperature	01/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot07:UI-16/TK01D-TE10_M
50	Air leaving temperature	01/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot07:UI-16/TK01D-TE30_M
51	TK return water temperature alt id	01/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot07:UI-16/TK01D-TE45_M
52	TK heating valve position	21/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot05:AO-V-8/TK01D-TV45_Y
53	TK air humidity	01/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot09:UI-16/TK01D-ME20_M
54	Air entering temperature	01/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot08:UI-16/TK02D-TE10_M
55	Air leaving Pressure	01/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot08:UI-16/TK02D-TE30_M
56	TK return water temperature	01/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot08:UI-16/TK02D-TE45_M

57	PF speed %	21/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot05:AO-V-8/TK02D-SC01_Y
58	TF speed %	21/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot05:AO-V-8/TK02D-SC02_Y
59	TK heating valve position	21/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot05:AO-V-8/TK02D-TV45_Y
60	Air leaving CO2	01/Tampere-Vuoreksen koulukeskus-AS5/IO Bus/Slot08:UI-16/TK02D-QIE30_M

Appendix 4: Code

Contents of data_processing.ipynb

Data processing

In [2]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from datetime import datetime as dt
from datetime import timedelta
import time
```

In [3]:

```
df_test = pd.read_csv('Vuores_dataset_17032021_19032021.csv', sep=';',
low_memory=False
)
```

In [4]:

```
df_big_test = pd.read_csv('dataset_updated_17_02_2021.csv', sep=',',
low_memory=False)
```

In [6]:

```
dwing_parameters = pd.read_excel('D-wing Vuores item_ids.xlsx')
```

In [18]:

```
def resample_dataframe(df, parameters, dt_format):
    resampled_df = pd.DataFrame()    for parameter in
 range(len(parameters['item_id'].values)):
 #print(parameters['item_id'].values[parameter])
 #print(parameters['Description'].values[parameter])
 item_id = parameters['item_id'].values[parameter]       item_id
 = item_id.replace('\xa0', ' ')       item_series =
 df[df['item_id']==item_id]       item_series =
 item_series[['inserted_at', 'value']]
        item_series['inserted_at'] = pd.to_datetime(item_series['inserted_at'], format=
dt_format)
        #item_series['inserted_at'] = pd.to_datetime(item_series['inserted_at'], format
='%Y-%m-%d %H:%M:%S')
 item_series.set_index('inserted_at', inplace=True)

        #indecies = item_series.index
        #indecies = indecies.drop_duplicates(keep='first')
        #item_series = item_series.loc[indecies]

        item_series = item_series.groupby(item_series.index).first()


        resampled_series = item_series.resample('H').pad().astype('float64')
  resampled_series.columns = [parameters['Description'].values[parameter]]
  resampled_df = pd.concat([resampled_df, resampled_series], axis=1)
  resampled_df.fillna(method='ffill')
    resampled_df = resampled_df.drop(resampled_df.head(1).index)
  resampled_df.dropna(inplace=True) #Approximately 1% of values is lost
        return
  resampled_df
```

In [20]:

```
resampled_df = resample_dataframe(df_big_test, dwing_parameters, '%Y-%m-%d %H:%M:%S') resampled_df_val = resample_dataframe(df_test, dwing_parameters, '%d/%m/%Y %H:%M:%S')
```

In [23]:

```
temp_cols = resampled_df_val.loc[:, 'Room temperature':'Room temperature'].astype('float64') co2_cols = resampled_df_val.loc[:, 'Room CO2':'Room CO2'].astype('float64')
```

In [24]:

```
resampled_df_val['Average room temperature'] = temp_cols.mean(axis=1)
resampled_df_val['Average room CO2'] = co2_cols.mean(axis=1)
resampled_df['Average room temperature'] = temp_cols.mean(axis=1)
resampled_df['Average room CO2'] = co2_cols.mean(axis=1)
#resampled_df['Average room temperature'] = resampled_df[['Room temperature']].mean(axi s=0)
```

In [25]:

```
modified_df_val = resampled_df_val.drop(['Room temperature'], axis=1) modified_df = resampled_df.drop(['Room temperature'], axis=1)
```

In [26]:

```
modified_df_val = modified_df_val.drop(['Room CO2'], axis=1)
modified_df = modified_df.drop(['Room CO2'], axis=1)
```

In [29]:

```
modified_df . to_csv ( 'Modified_df.csv'    )
modified_df_val  . to_csv ( 'Modified_df_val.csv'     )
```

Contents of feature_selection_and_model.ipynb

Feature Selection

In [ ]:

```
# import all libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import scale
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import GridSearchCV
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_regression
from sklearn.metrics import mean_absolute_error
from sklearn.model_selection import RepeatedKFold
from sklearn.pipeline import Pipeline
from sklearn.feature_selection import mutual_info_regression



import warnings# supress warnings
warnings . filterwarnings    ( 'ignore'   )
```

In [ ]:

```
vuores = pd.read_csv('Modified_df.csv') vuores = vuores.iloc[:,1:].astype('float64')
vuores_val = pd.read_csv('Modified_df_val.csv') vuores_val =
vuores_val.iloc[:,1:].astype('float64') vuores_columns = vuores.columns.to_list()
```

In [ ]:

```
sns.regplot(x="Outside temperature 1 **Used to control**", y="Energy PV03", data=vuores
[['Energy PV03', "Outside temperature 1 **Used to control**"]], fit_reg=False)
```

In [ ]:

```
sns.regplot(x="Outside temperature 1 **Used to control**", y="Average room temperature"
, data=vuores[['Average room temperature', "Outside temperature 1 **Used to control**"
]], fit_reg=False)
```

In [ ]:

```
sns.regplot(x="Outside temperature 1 **Used to control**", y="Average room CO2", data=v
uores[['Average room CO2', "Outside temperature 1 **Used to control**"]], fit_reg=False )
```

6 Feature selection for Energy PV03

```
vuores_columns = vuores.columns.to_list()

y_columns = vuores_columns.pop(vuores_columns.index('Energy PV03'))
X_columns = vuores_columns

y_energy = vuores[y_columns]

X_energy = vuores[X_columns]

X_train, X_test, y_train, y_test = train_test_split(X_energy, y_energy, test_size=0.2,
random_state=1)

scalery_energy = MinMaxScaler().fit(y_train.values.reshape(-1, 1))
scalerX_energy = MinMaxScaler().fit(X_train)
X_train = scalerX_energy.transform(X_train)
y_train = scalery_energy.transform(y_train.values.reshape(-1, 1))
X_test = scalerX_energy.transform(X_test)
y_test = scalery_energy.transform(y_test.values.reshape(-1, 1))
X_scaled = scalerX_energy.transform(X_energy)
y_scaled = scalery_energy.transform(y_energy.values.reshape(-1, 1))

def select_featuresB(X_train, y_train, X_test, k):

    fs = SelectKBest(score_func=f_regression, k=k)
    fs.fit(X_train, y_train)
    X_train_fs = fs.transform(X_train)
    X_test_fs = fs.transform(X_test)
    return X_train_fs, X_test_fs, fs

X_train_fs, X_test_fs, fs = select_featuresB(X_train, y_train, X_test, 'all')
scores_dict = {}
for i in range(len(fs.scores_)):
    print('Feature %d: %f' % (i, fs.scores_[i]), X_columns[i])
    scores_dict[X_columns[i]] = fs.scores_[i]
plt.bar([i for i in range(len(fs.scores_))], fs.scores_)
plt.show()

# evaluate the model
yhat = model.predict(X_test)
# evaluate predictions
mae = mean_absolute_error(y_test, yhat)
print('MAE: %.3f' % mae)
```

cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)

model = LinearRegression()

fs = SelectKBest(score_func=f_regression)

pipeline = Pipeline(steps=[('sel',fs), ('lr', model)])

```
grid = dict()
grid['sel__k'] = [i for i in range(X_scaled.shape[1]-20, X_scaled.shape[1]+1)]
# define the grid search
search = GridSearchCV(pipeline, grid, scoring='neg_mean_squared_error', n_jobs=-1,
cv=c v)
# perform the search
results_energy = search.fit(X_scaled, y_scaled)
# summarize best
print('Best MAE: %.3f' % results_energy.best_score_) print('Best
Config: %s' % results_energy.best_params_)
# summarize all
means =
results_energy.cv_results_['mean_test_score']
params = results_energy.cv_results_['params'] for
mean, param in zip(means, params):    print(">%.3f
with: %r" % (mean, param)) In [ ]:
```

```
best_energy  = results_energy  . best_estimator_

X_val_scaled   = scalerX_energy  . transform ( vuores_val  [ X_columns])
X_val_scaled   = pd. DataFrame( X_val_scaled  )
X_val_scaled  . columns  = X_columns
display  ( y_columns)

y_pred_energy  = best_energy . predict  ( X_val_scaled  . values )
y_pred_energy  = [ item  for sublist   in y_pred_energy  for item  in sublist  ]


y_pred_energy  = scalery_energy  . inverse_transform  ([ y_pred_energy ])
y_pred_energy  = [ item  for sublist   in y_pred_energy  for item  in sublist  ]


y_val_energy   = vuores_val  [ y_columns] . values

plt . plot ( n,  y_pred_energy )
plt . plot ( n,  y_val_energy  )
plt . ylabel  ( 'Energy PV03'  )
plt . xlabel  ( 'At validation observation №'       )

mae = mean_absolute_error ( y_val_energy ,  y_pred_energy )
mae_lr  = mean_absolute_error ( y_pred_energy_lr  ,  y_val_energy  )
print  ( 'MAE for Energy PV03:'    , mae)
plt . show()
```

7 Feature selection for Average room temperature

```
vuores_columns = vuores.columns.to_list()
parameter = 'Average room temperature'
y_columns = vuores_columns.pop(vuores_columns.index(parameter))
X_columns = vuores_columns

y_temp = vuores[y_columns]
X_temp = vuores[X_columns]

X_train, X_test, y_train, y_test = train_test_split(X_temp, y_temp, test_size=0.2, rand
om_state=1)

scalery_temp = MinMaxScaler().fit(y_train.values.reshape(-1, 1))
scalerX_temp = MinMaxScaler().fit(X_train) X_train_temp =
scalerX_temp.transform(X_train)
y_train_temp = scalery_temp.transform(y_train.values.reshape(-1, 1)) X_test_temp
= scalerX_temp.transform(X_test)
y_test_temp = scalery_temp.transform(y_test.values.reshape(-1, 1))
X_scaled_temp = scalerX_temp.transform(X_temp)
y_scaled_temp = scalery_temp.transform(y_temp.values.reshape(-1, 1))

def select_featuresB(X_train, y_train, X_test):

    fs = SelectKBest(score_func=f_regression, k='all')
    fs.fit(X_train, y_train)
    X_train_fs = fs.transform(X_train)
X_test_fs = fs.transform(X_test)    return
X_train_fs, X_test_fs, fs scores_dict = {}

X_train_fs, X_test_fs, fs = select_featuresB(X_train, y_train, X_test) for
i in range(len(fs.scores_)):    print('Feature %d: %f' % (i, fs.scores_[i]),
X_columns[i])    scores_dict[X_columns[i]] = fs.scores_[i]
plt.bar([i for i in range(len(fs.scores_))], fs.scores_) plt.show()
# evaluate the model yhat
= model.predict(X_test) #
evaluate predictions
```

```
mae = mean_absolute_error(y_test, yhat)
print('MAE: %.3f' % mae)
```

```
cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)

model = LinearRegression()
#model = RandomForestRegressor()
fs = SelectKBest(score_func=f_regression)
pipeline = Pipeline(steps=[('sel',fs), ('lr', model)])


grid = dict()
grid['sel__k'] = [i for i in range(X_scaled_temp.shape[1]-20, X_scaled_temp.shape[1]+1
)]
# define the grid search
search = GridSearchCV(pipeline, grid, scoring='neg_mean_squared_error', n_jobs=-1, cv=c
v)
# perform the search
results_temp = search.fit(X_scaled_temp, y_scaled_temp)
# summarize best
print('Best MAE: %.3f' % results_temp.best_score_)
print('Best Config: %s' % results_temp.best_params_)
# summarize all
means = results_temp.cv_results_['mean_test_score']
params = results_temp.cv_results_['params']
for mean, param in zip(means, params):
    print(">%.3f with: %r" % (mean, param))
```

In [ ]:

```
best_temp = results_temp . best_estimator_

X_val_scaled    = scalerX_temp . transform ( vuores_val [ X_columns])
y_val_temp  = vuores_val [ y_columns]

y_pred_temp = best_temp. predict ( X_val_scaled  )

y_pred_temp = scalery_temp . inverse_transform  ( y_pred_temp reshape ( - 1,  1))


t  = range( len ( y_pred_temp))
mae = mean_absolute_error ( y_pred_temp,  y_val_temp)
mae_lr  = mean_absolute_error ( y_pred_temp_lr ,  y_val_temp)
plt . plot ( t ,  y_pred_temp flatten  ( 'C'  ))
plt . plot ( t ,  y_val_temp)
print  ( 'MAE for Average room temperature:'     , mae)
plt . ylabel  ( 'Average room temperature'   )
plt . xlabel  ( 'At validation observation №'       )
plt . show()
```

8 Feature selection for Average room CO2

```
vuores_columns = vuores.columns.to_list()
parameter = 'Average room CO2'
y_columns = vuores_columns.pop(vuores_columns.index(parameter))
X_columns = vuores_columns

y_CO2 = vuores[y_columns]

X_CO2 = vuores[X_columns]

X_train, X_test, y_train, y_test = train_test_split(X_CO2, y_CO2, test_size=0.2, random
_state=1)

scalery = MinMaxScaler().fit(y_train.values.reshape(-1, 1))
scalerX = MinMaxScaler().fit(X_train)
X_train = scalerX.transform(X_train)
y_train = scalery.transform(y_train.values.reshape(-1, 1))
X_test = scalerX.transform(X_test)
y_test = scalery.transform(y_test.values.reshape(-1, 1))
X_scaled = scalerX.transform(X_CO2)
y_scaled = scalery.transform(y_CO2.values.reshape(-1, 1))

def select_featuresB(X_train, y_train, X_test):

    fs = SelectKBest(score_func=f_regression, k='all')
    fs.fit(X_train, y_train)
    X_train_fs = fs.transform(X_train)
    X_test_fs = fs.transform(X_test)
    return X_train_fs, X_test_fs, fs


scores_dict = {}

X_train_fs, X_test_fs, fs = select_featuresB(X_train, y_train, X_test)
for i in range(len(fs.scores_)):
    print('Feature %d: %f' % (i, fs.scores_[i]), X_columns[i])
    scores_dict[X_columns[i]] = fs.scores_[i]
plt.bar([i for i in range(len(fs.scores_))], fs.scores_)
plt.show()
```

```
cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)

model = LinearRegression()
fs = SelectKBest(score_func=f_regression)
pipeline = Pipeline(steps=[('sel',fs), ('lr', model)])


grid = dict()
grid['sel__k'] = [i for i in range(X_scaled.shape[1]-20, X_scaled.shape[1]+1)]
```

```
# define the grid search
search = GridSearchCV(pipeline, grid, scoring='neg_mean_squared_error', n_jobs=-1,
cv=c v)
# perform the search
results_CO2 = search.fit(X_scaled, y_scaled)
# summarize best
print('Best MSE: %.3f' % results_CO2.best_score_) print('Best
Config: %s' % results_CO2.best_params_)
# summarize all
means =
results_CO2.cv_results_['mean_test_score']
params = results_CO2.cv_results_['params'] for
mean, param in zip(means, params):
print(">%.3f with: %r" % (mean, param)) In [ ]:
```

```
best_CO2 = results_CO2 . best_estimator_

X_val_scaled    = scalerX  . transform ( vuores_val  [ X_columns])
y_val_CO2  = vuores_val  [ y_columns]

y_pred_CO2 = best_CO2. predict  ( X_val_scaled   )

y_pred_CO2 = scalery  . inverse_transform   ( y_pred_CO2 reshape ( - 1,   1))

t  = range( len ( y_pred_CO2))
mae = mean_absolute_error ( y_pred_CO2,  y_val_CO2)
mae_lr  = mean_absolute_error ( y_pred_CO2_lr ,  y_val_CO2)
plt . plot ( t ,  y_pred_CO2 flatten  ( 'C'  ))
plt . plot ( t ,  y_val_CO2)
plt . ylabel  ( 'Average room CO2'  )
plt . xlabel  ( 'At validation observation №'       )
print  ( 'MAE for average room CO2:'     , mae)
```

In [ ]: