Udacity Capstone project

Integrating Machine Learning into Model Predictive Control

Project overview

Introduction:

Objectives/Scope:

Model Predictive Control (MPC) is a popular control approach that optimisms control action across a finite time horizon by using a predictive model of the system. The predictive model is crucial for the control strategy's success, and traditional modelling approaches frequently rely on first-principles models, which might be constrained by simplifying assumptions and uncertainties in the parameters and inputs. Machine learning is a strong method that can increase the predictive model's accuracy, efficiency, and resilience in MPC for non-linear systems. This study investigates the problems and opportunities associated with applying machine learning in MPC and provides a review of cutting-edge methodologies and methods.

Methods, Procedures, Process:

Several machine learning algorithms were used to construct a model for use in model predictive control (MPC) in this study. Our model was constructed utilising Long Short-Term Memory (LSTM), TabTransformers, Gated Recurrent Units (GRU), Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN).After gathering and preprocessing data from the controlled system. This included information on the system's inputs, outputs, and any external factors that could influence its behaviour. The data is then separated into training and testing collections. Next, we trained individual models using each of the aforementioned machine learning algorithms. We trained each model using the training data and evaluated their performance using the testing data. We compared the performance of each model and chose the model with the best results for use in MPC.

Domain

A sophisticated form of process control called model predictive control (MPC) is used to manage a process while adhering to a set of restrictions. (1)Since the 1980s, it has been utilized in chemical and oil refineries as well as process industries. (2)It has recently been employed in power electronics, models for balancing power systems and automotive industry(3). Model predictive controllers rely on dynamic processes models, most frequently linear empirical models acquired by system identification. The fundamental benefit of MPC is that it enables timeslot optimization while taking future timeslots into consideration. MPC is also capable of foreseeing future events and taking appropriate management measures. This prediction capability is not present in PID controllers.(4)

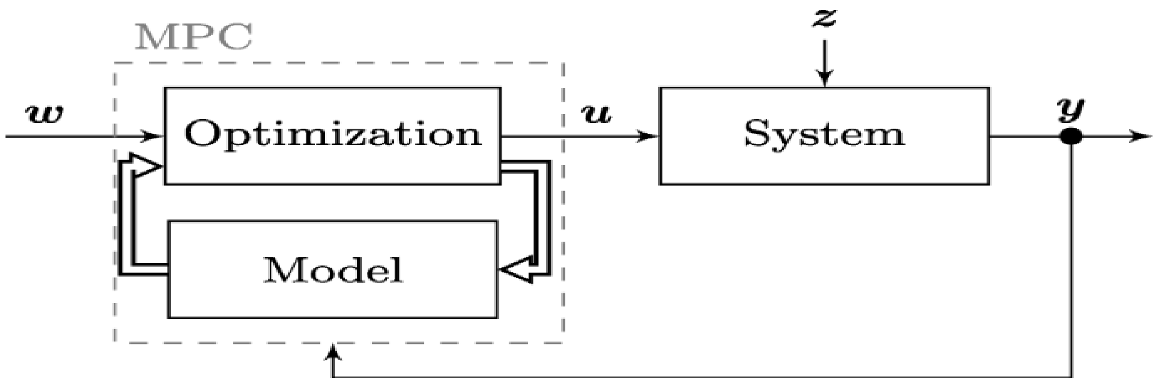


Figure.1.1 general MPC configueration

Problem statement

Control is one of several sectors where machine learning has lately become successful and gained popularity. Machine learning is a set of techniques for extracting mathematical models from data. (5)

Machine Learning (ML) can be used to resolve the following problems in the field of MPC:

How can one create accurate prediction models from data?

How can I choose the ideal MPC parameters?

Can the problem be made smaller?

How do you warm up the solver?

Datasets and Inputs:

The data set are obtained from a fractionator tower real data during step testing process which is used to implement MPC control scheme to the plant where the fractions of (Propane (C_3H_8), Pentane (C_5H_{12}) and Heptane (C_7H_{16})) are separated. The distributed control system (DCS), advanced regulatory controllers (ARCs), and laboratory data are the sources of inputs during the test.

Number of samples=5820 Number of variables (Tags)=18 Sample period=60 sec.

Time range for the dataset: From: 10/01/2009 8:14:00 To:10/05/2009 9:13:00

Dataset variables:

AI-2020	MOL	OVERHEAD C5'S
AI-2021	MOL	MIDDLE C7'S
AI-2022	MOL	BOTTOM C3'S
FIC-2100PV	SCFH	FEED FURNACE FUEL
FIC-2101PV	MBBL/D	TOP PRODUCT
FIC-2102PV	MBBL/D	BOTTOM PRODUCT
FI-2005PV	MBBL/D	Feed Flow
FIC-2001SP	MBBL/D	TOP REFLUX SETPOINT
FIC-2001OP	%	TOP REFLUX OUTPUT

FIC-2001PV	MBBL/D	TOP REFLUX SETPOINT
FIC-2002SP	MBBL/D	MIDDLE PRODUCT DRAW SETPOINT
FIC-2002OP	%	MIDDLE PRODUCT DRAW OUTPUT
FIC-2002PV	MBBL/D	MIDDLE PRODUCT DRAW
FIC-2004SP	MBBL/D	MIDDLE REFLUX SETPOINT
FIC-2004OP	%	MIDDLE REFLUX OUTPUT
FIC-2004PV	MBBL/D	TOP REFLUX
QI-2106PV	BTU/H	MIDDLE REFLUX DUTY
TIC-2003SP	DEG F	FEED TEMPERATURE SETPOINT

Solution statement

In this project we will try to address the first problem where several models will be built and their accuracy for prediction will be tested against real data obtained from step testing for fractionator plant where manipulated variables of the system are changed to estimate and the changed in the controlled variables of the system are measured. By stabilizing operation, boosting throughput, enhancing fractionator performance, lowering product quality loss, and lowering utility consumption, model-predictive control (MPC) enhances the capability of process units.

Furthermore, higher-level applications like planning models and process optimizers can use real-time data from MPC.

The solution for this problem is obtained by applying one of the following models:

- FIR (finite impulse response) Model
- State space Model

The models can be calculated using commercial grade programs like (ASPEN DMC3) or even by using MATLAB MPC toolbox.

for the MPC. A well-designed MPC controller moves several variables simultaneously once every minute, or even more often in some circumstances, in response to any disturbance variables present to the system like changes in feedstock, ambient temperature, and other factors.

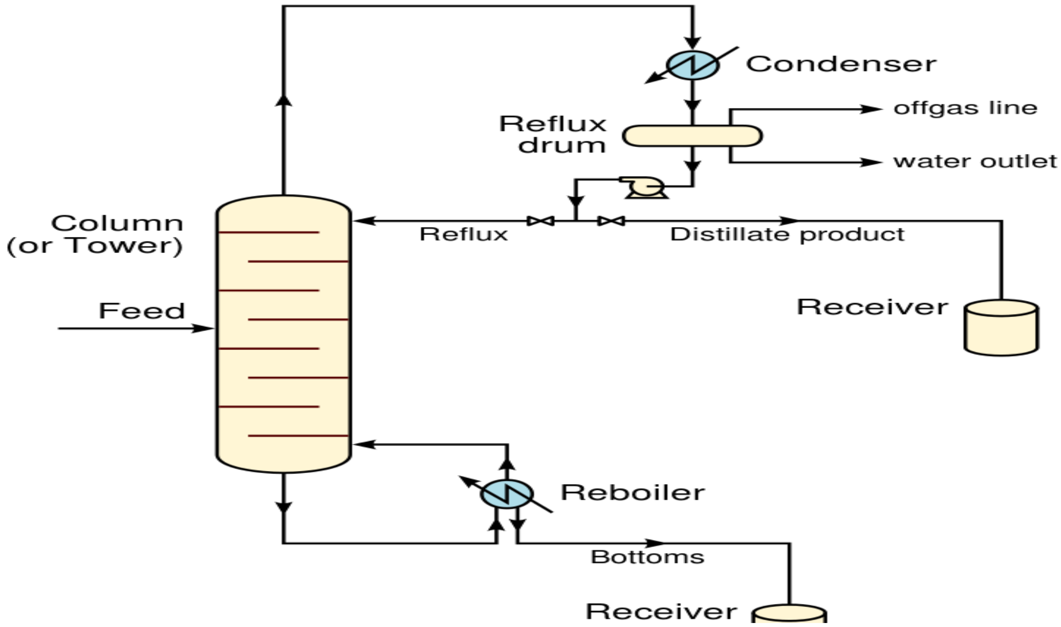


Figure 1.2 General fractionator tower

Evaluation Metrics

Since we are dealing with nonlinear regression model several metrics will be used to determine the accuracy of our model where each metric has ups and downs:

RMSE:

Root Mean Squared Error (This is just the square root of the MSE.) eliminating MSE from choice as it more superior and related to the standard deviation of the error term; easy to calculate; in the same units of yet the relationship to standard deviation can range from unhelpful to downright misleading if the error is not Gaussian or does not have a constant standard deviation.

R2:

Although R2 is related to comparing your predictions to the predictions of a baseline model yet it has poorly performance when it comes to nonlinear models and lacks its usual "proportion of variance explained" interpretation.

MAPE: Mean Absolute Percentage Error

Handles data on different scales but overestimates and underestimates are not penalized equally.

Data Analysis

Data Exploration and visualization:

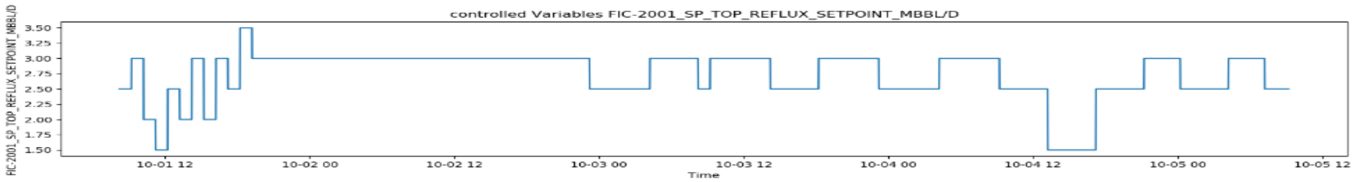
the note book (data preparation) will be a starter file for the data preparation, data loading, data cleaning, data visualization, feature engineering and uploading train and test datasets to S3 to be used in the model training pipeline and to be used in the model deployment pipeline.

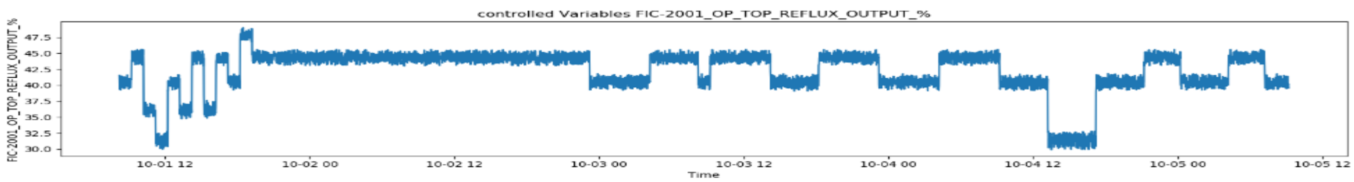
- In this file data preparation for the first and second project was carried out
- Data cleaning by removing outliers from dataset.
- Data augmentation for the target to be predicted variables to form time series dataset to prepare time series forecasting.

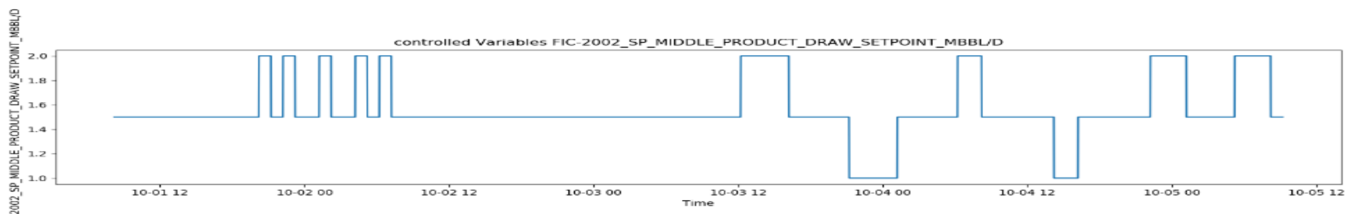
Data visualization:

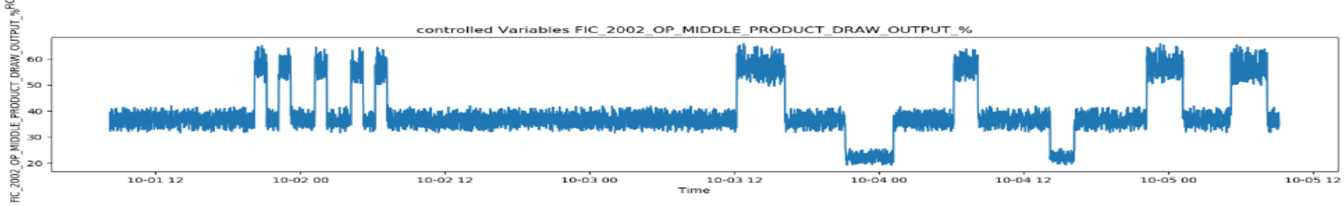
The data visualization for controlled variables:

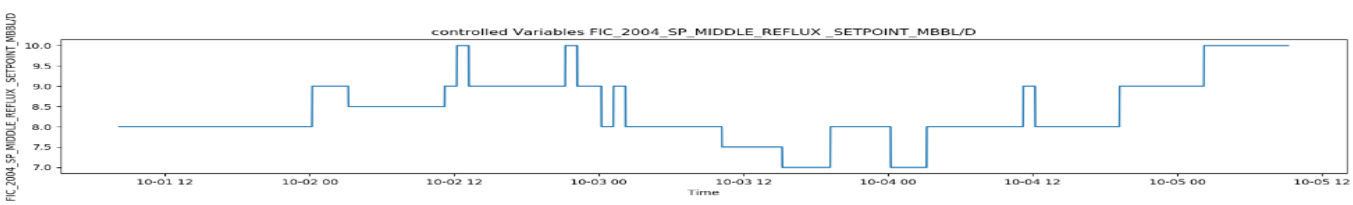
Data visualization:

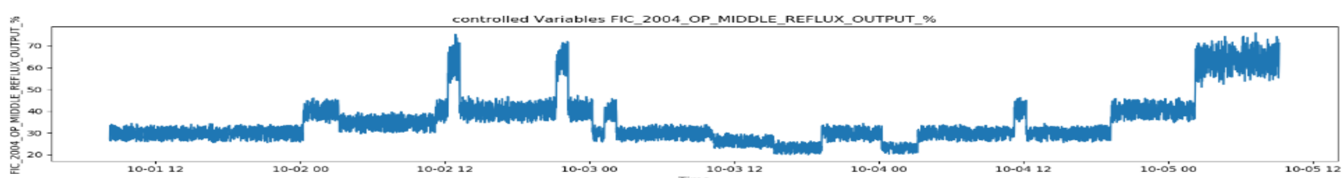


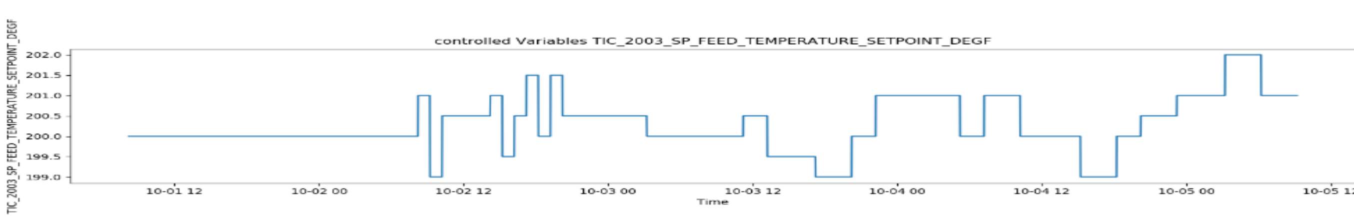




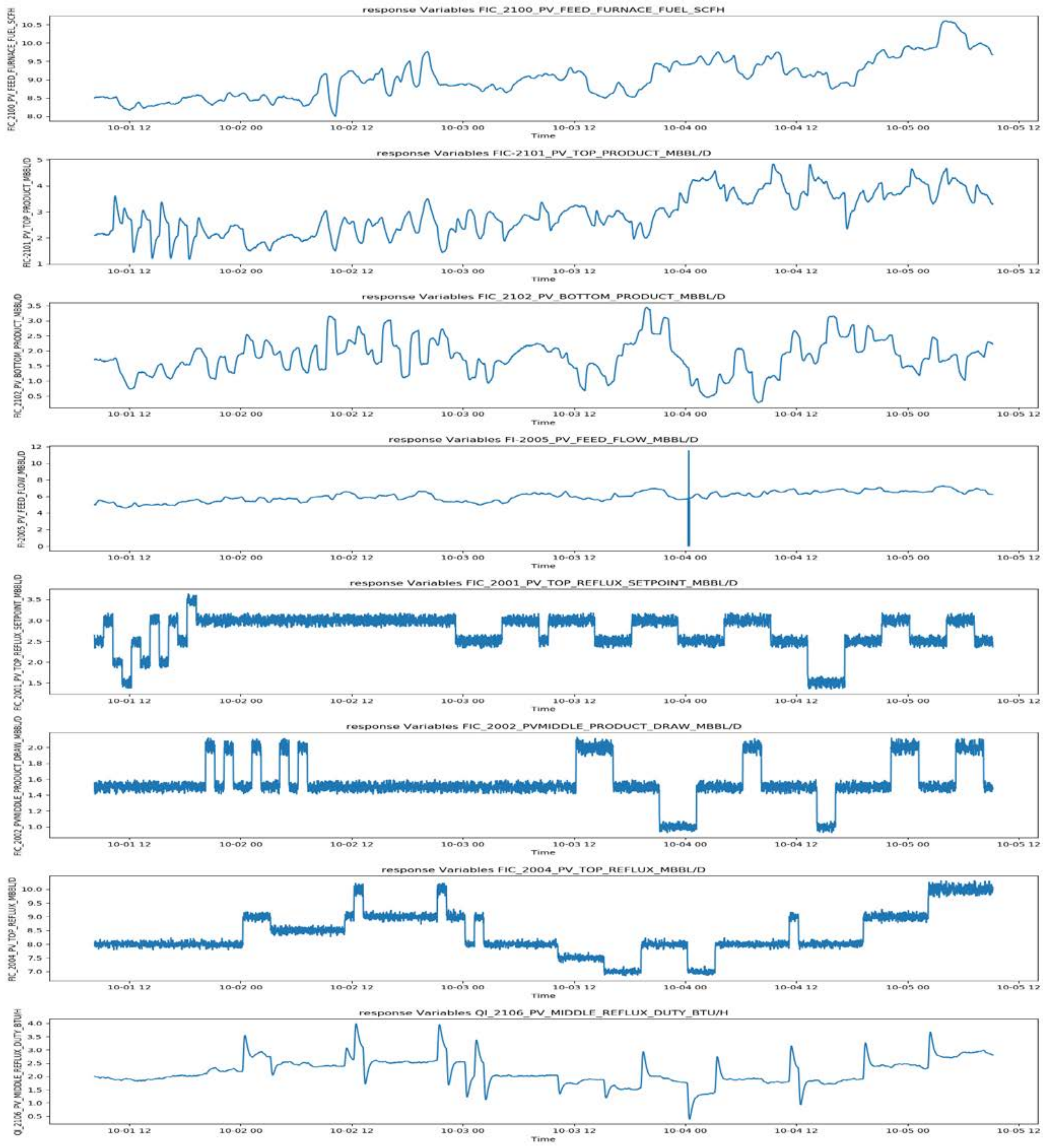




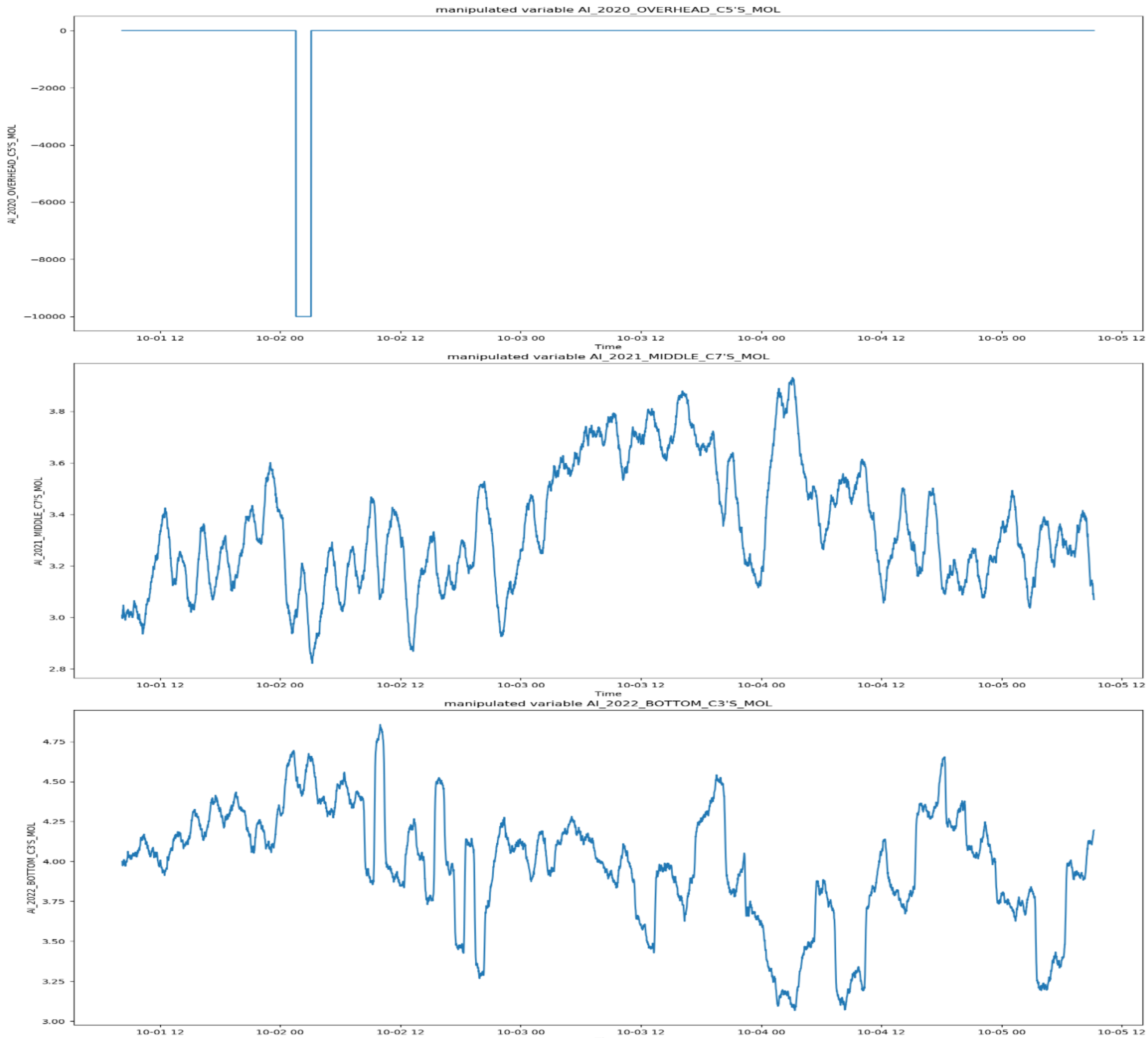




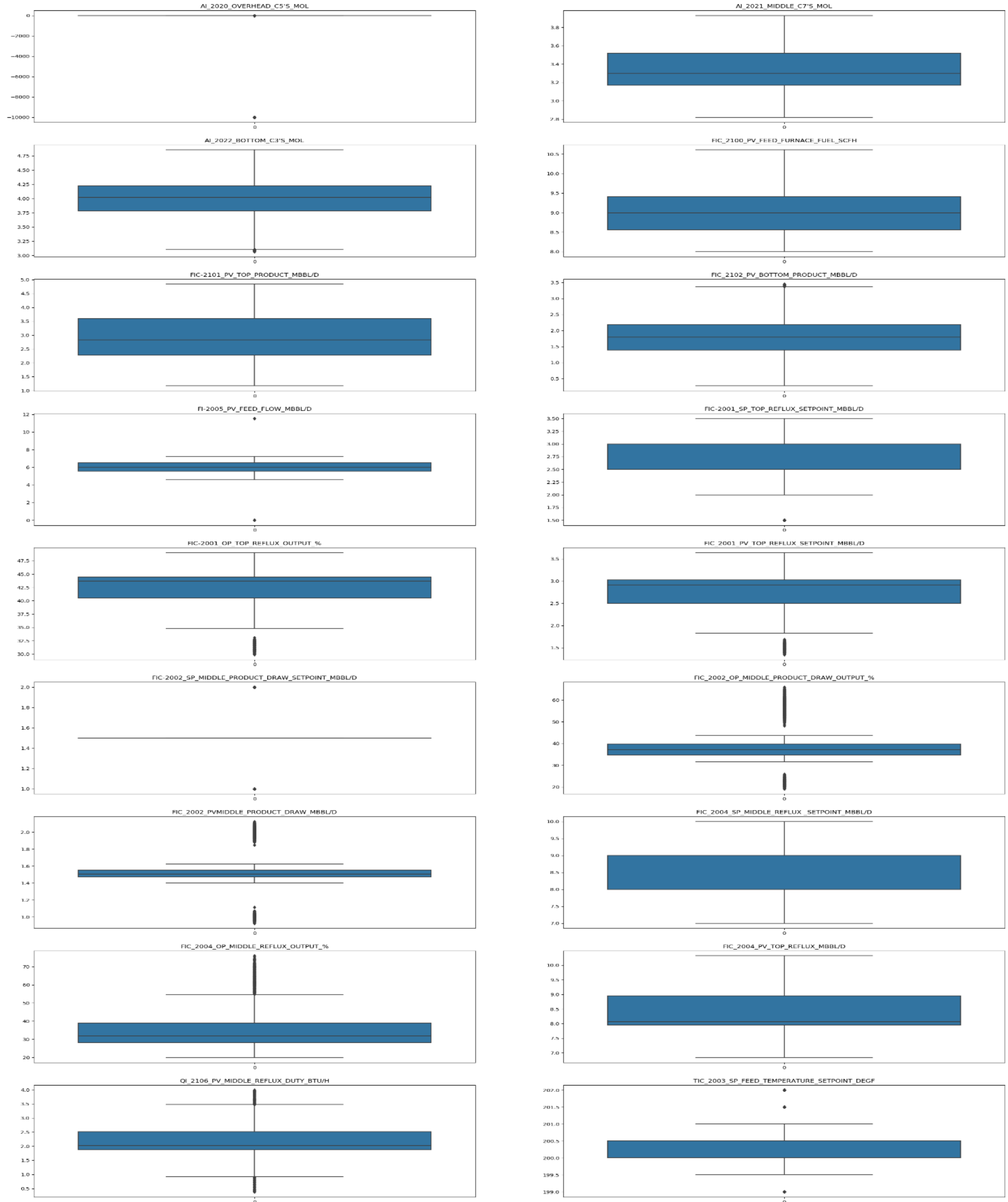
The data visualization for response variables:



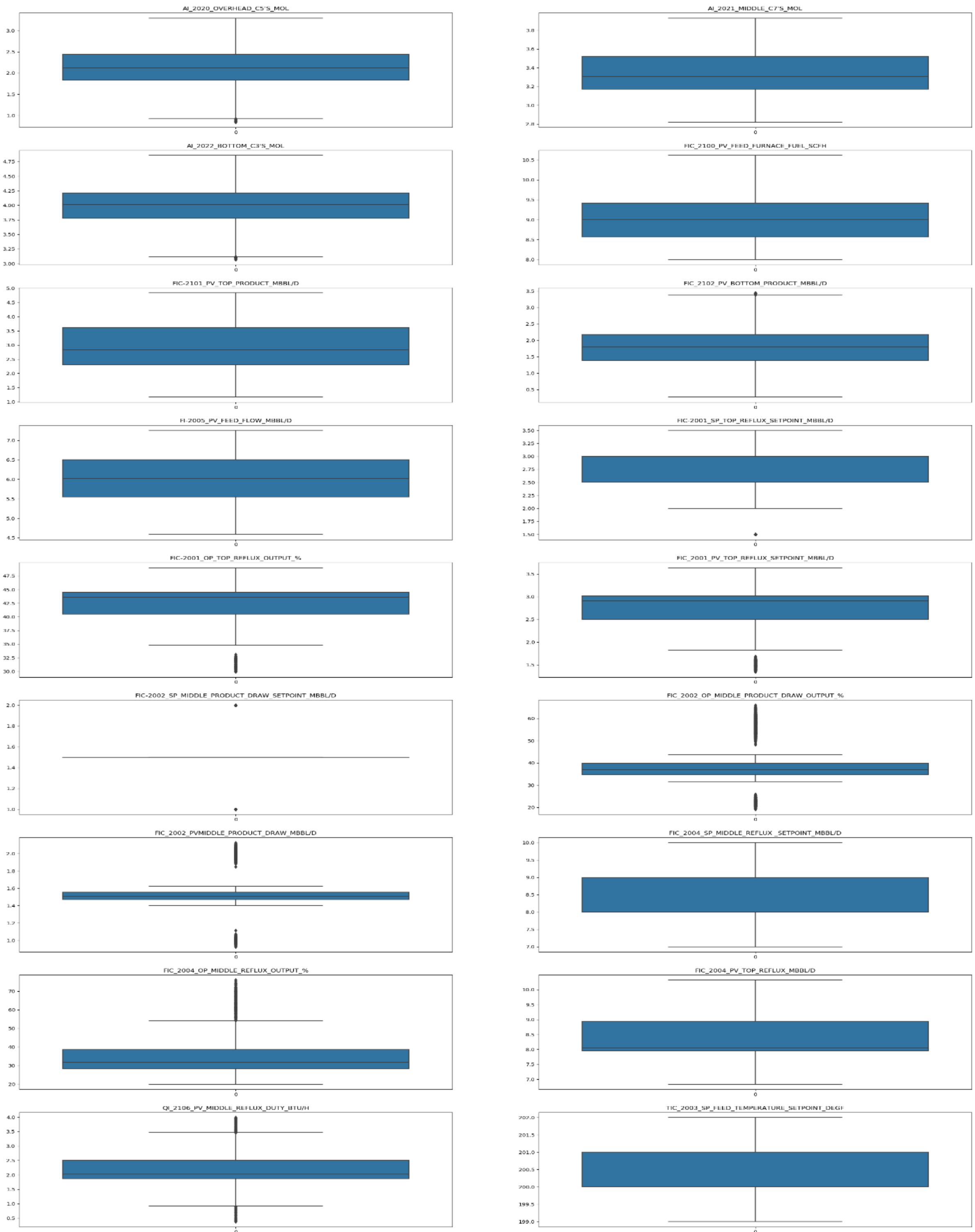
Data visualization for manipulated variables:



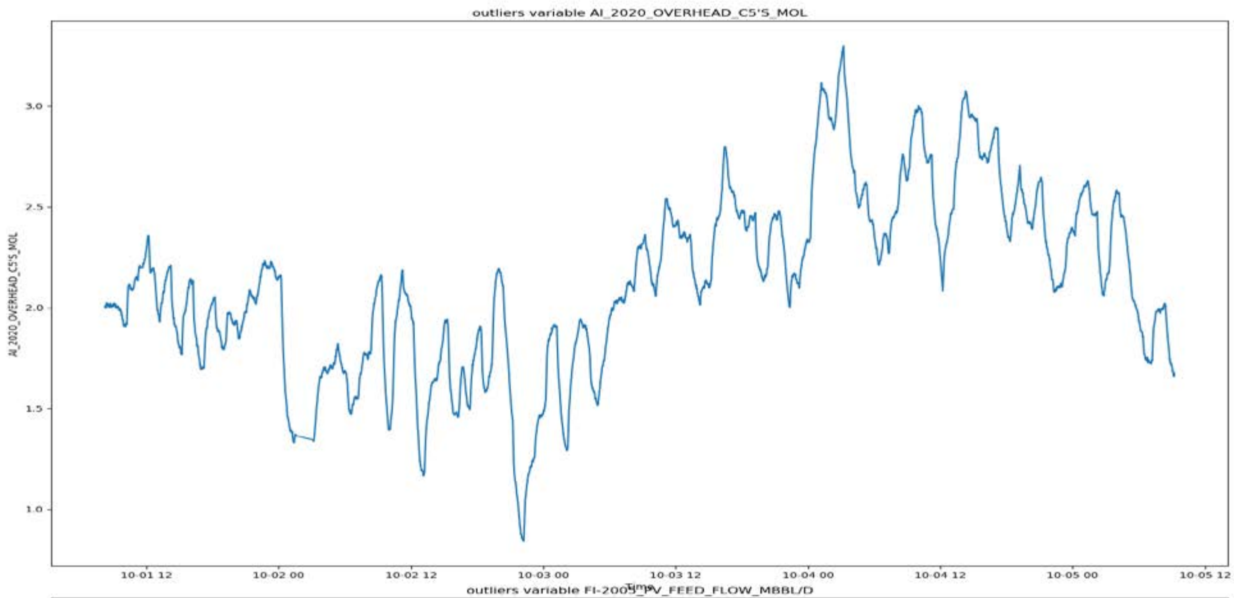
To detect outliers in the data set boxplot for the dataset will be used:

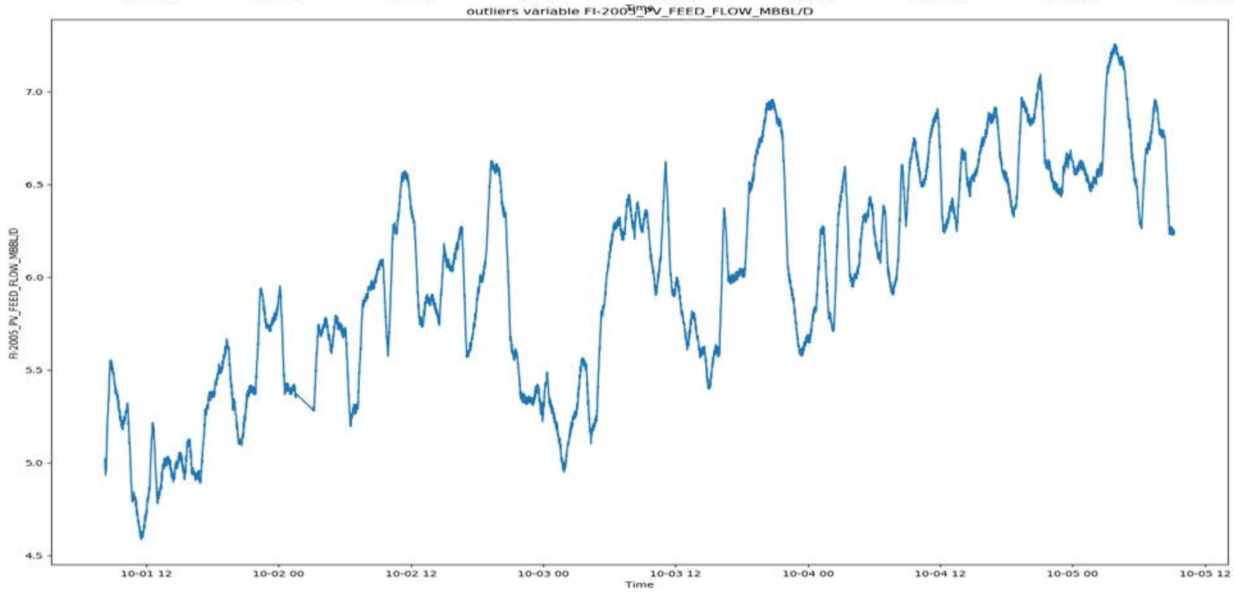


After removing the outlies:



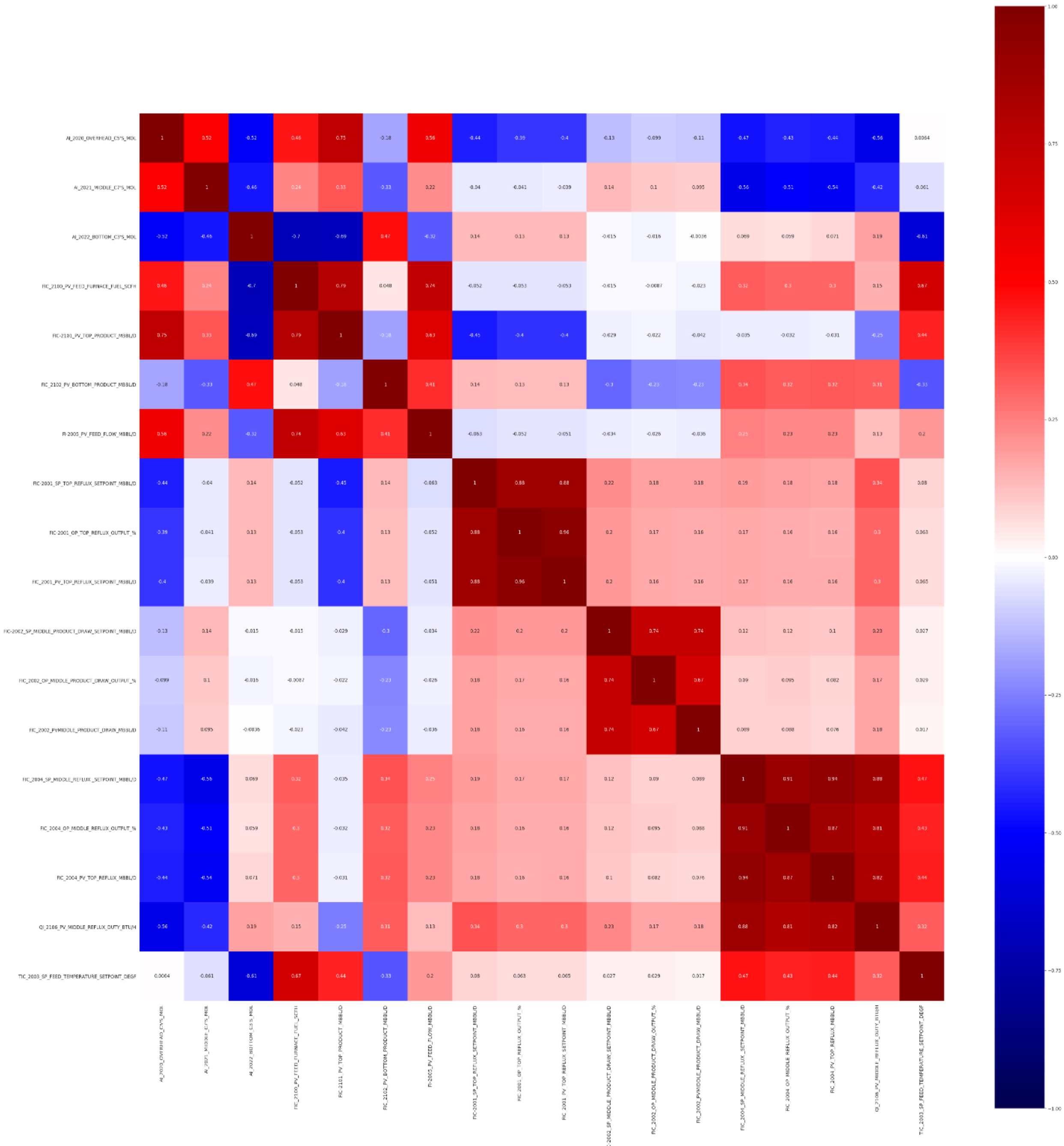
Plotting the variables after fixing outliers:



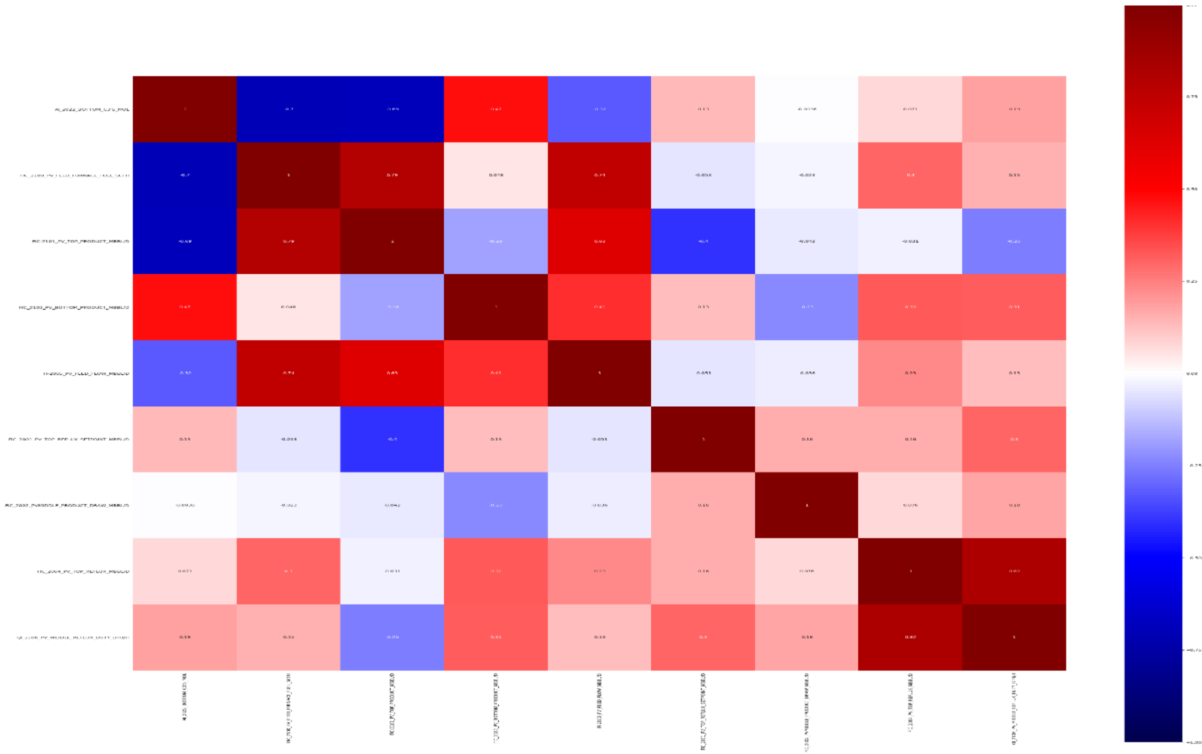


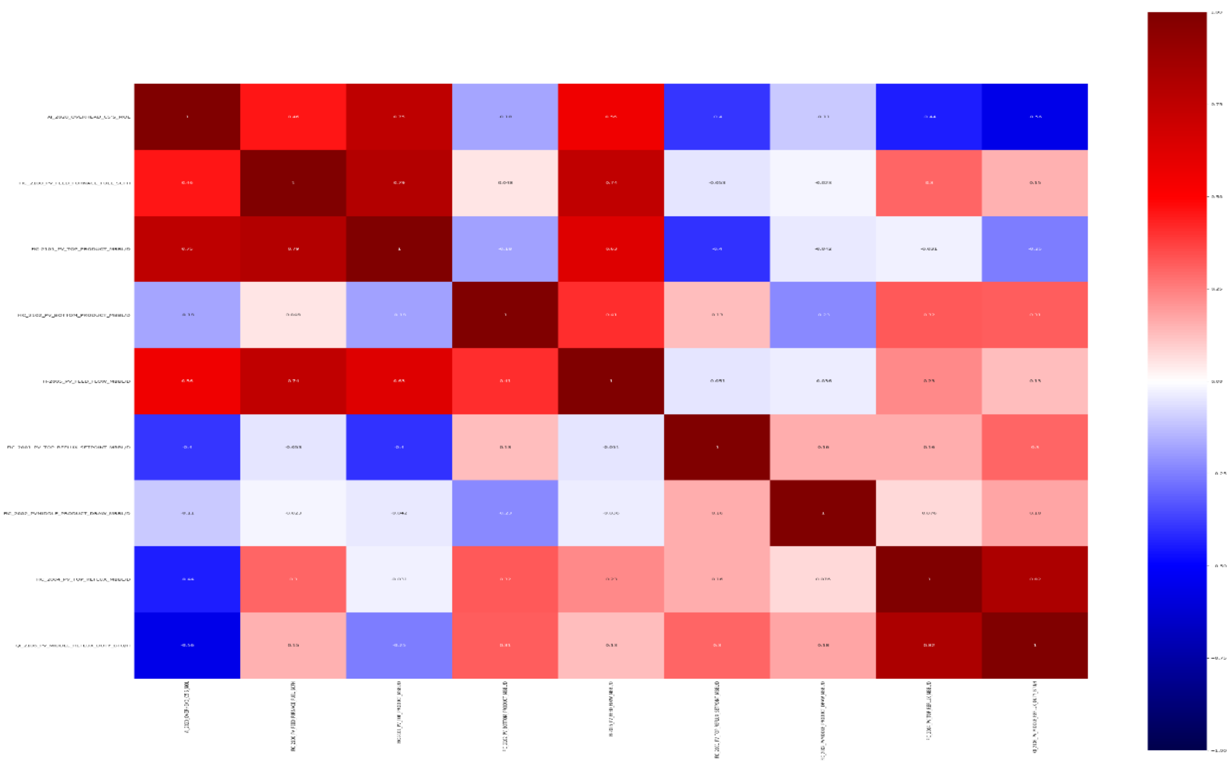
Data correlation:

To check for the correlation between data we used spearman correlation method for visualization heat map was used. For the whole dataset and for the manipulated and controlled Variables individually.



AI_2020_OVERHEAD_C5'S_MOL
AI_2021_MIDDLE_C7'S_MOL
AI_2022_BOTTOM_C3'S_MOL
FIC_2100_PV_FEED_FURNACE_FUEL_SCFH
FIC-2101_PV_TOP_PRODUCT_MBBL/D
FIC_2102_PV_BOTTOM_PRODUCT_MBBL/D
FI-2005_PV_FEED_FLOW_MBBL/D
FIC-2001_SP_TOP_REFLUX_SETPOINT_MBBL/D
FIC-2001_OP_TOP_REFLUX_OUTPUT_%
FIC_2001_PV_TOP_REFLUX_SETPOINT_MBBL/D
FIC-2002_SP_MIDDLE_PRODUCT_DRAW_SETPOINT_MBBL/D
FIC_2002_OP_MIDDLE_PRODUCT_DRAW_OUTPUT_%
FIC_2002_PVMIDDLE_PRODUCT_DRAW_MBBL/D
FIC_2004_SP_MIDDLE_REFLUX_SETPOINT_MBBL/D
FIC_2004_OP_MIDDLE_REFLUX_OUTPUT_%
FIC_2004_PV_TOP_REFLUX_MBBL/D
QI_2106_PV_MIDDLE_REFLUX_DUTY_BTU/H
TIC_2003_SP_FEED_TEMPERATURE_SETPOINT_DEGF
1.00
0.75
0.50
0.25
0.00
-0.25
-0.50
-0.75
-1.00



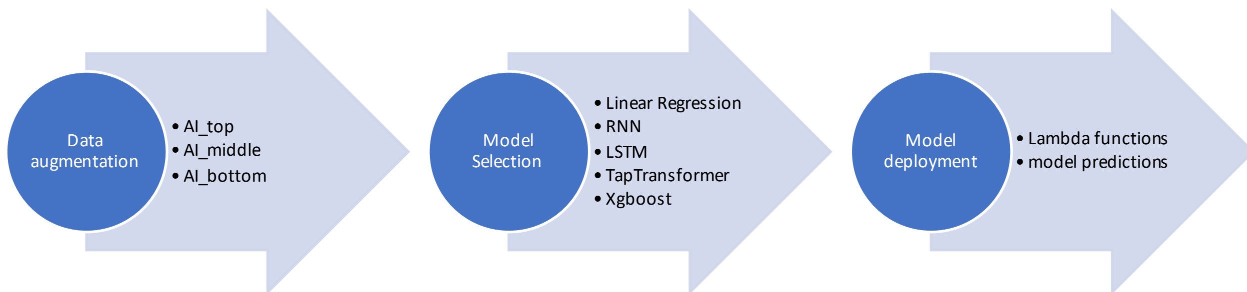


Methodology:

For this project two steps are implemented:

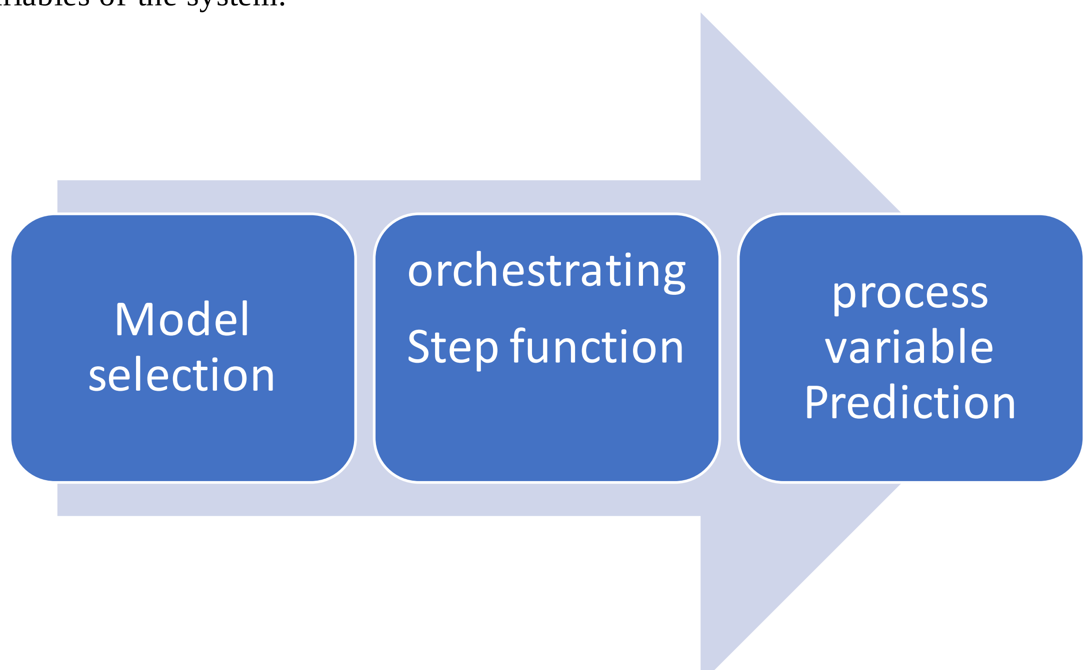
In the first step:

A time series forecasting model is built to predict the system analyzers using AWS environments lambda function will trigger the deployed models to predict the next values of system analyzer.



In the second step

The predicted system analyzer values will be used to predict the next values of the process variables of the system.



Data preprocessing:

First step:

For each analyzer (AI_top, AI_middle, AI_bottom) in the data set time series forecasting was applied for that data augmentation for each variable was done with time window of 10 steps to predict the following value then the data saved in csv file and stored in S3 bucket to be used in training the model.
Then the data were split into train, test and validation using splitting index with out shuffling to ensure the data dependency on time.
Sample of AI_bottom data for training with t10 as target value and from t0 to t9 are features.

t10	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9
3.99974	4	3.99831	3.97746	3.97262	3.98841	3.99264	3.994	3.99537	3.99177	3.99605
4.01184	3.99831	3.97746	3.97262	3.98841	3.99264	3.994	3.99537	3.99177	3.99605	3.99974
3.99499	3.97746	3.97262	3.98841	3.99264	3.994	3.99537	3.99177	3.99605	3.99974	4.01184
3.97706	3.97262	3.98841	3.99264	3.994	3.99537	3.99177	3.99605	3.99974	4.01184	3.99499
3.99156	3.98841	3.99264	3.994	3.99537	3.99177	3.99605	3.99974	4.01184	3.99499	3.97706

Benchmark Model

The Benchmark model that is used is linear regression models from scikit-learn library.

Technique	**MSE first step**	**MSE Second Step**
Linear Regressor	0.00616	0.5152606
Ridge Regressor	0.007185	0.4816907

Benchmark comparison findings:

Time series forecasting and regression methods LSTM and XGBoost are prominent. LSTM and XGBoost have advantages over linear regression:

LSTM and XGBoost can detect non-linear data correlations that linear regression cannot. In time series forecasting and regression, complicated non-linear connections between variables are crucial.

LSTM and XGBoost are designed to handle time series data, unlike linear regression. Seasonality and autocorrelation in time series data require specialised algorithms to model and forecast.

LSTM and XGBoost are more resilient to outliers than linear regression. LSTM and XGBoost are designed to address outliers, which can affect linear regression models.

Accuracy: LSTM and XGBoost generate accurate predictions. They combine ensemble learning and deep learning to increase prediction accuracy.

LSTM and XGBoost have many advantages over linear regression for time series forecasting and regression. They manage time series data, non-linear relationships, outliers, and make more accurate forecasts. The algorithm chosen relies on the task and data. To find the optimal algorithm for a problem, experiment and assess.

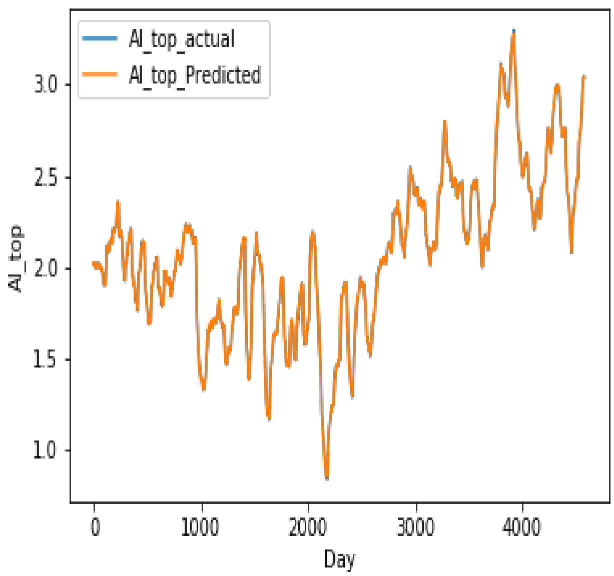
Implementation and Refinement:

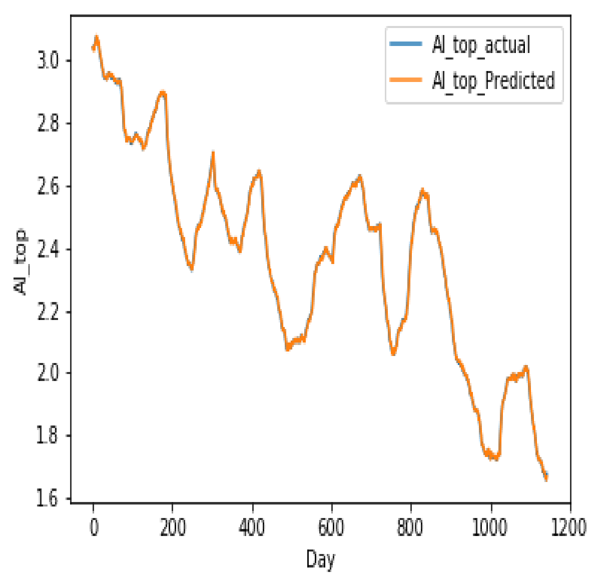
Forecasting time series is an important application of machine learning that has received a lot of attention in recent years due to the significant research that has been done on it. For time series forecasting, a wide variety of machine learning methods, such as linear regression, recurrent neural networks (RNNs), convolutional neural networks (CNNs), long short-term memory (LSTM) networks, TapTransformer, and XGBoost, have been offered as possible solutions. In this study, we evaluate the effectiveness of various methods on a time series forecasting problem and compare their results.
The following table summarize the algorithm against the evaluation metric:

Technique	**MSE**
Linear Regressor	0.00616
Ridge Regressor	0.007185
RNN	0.0049
CNN	0.00523
LSTM	0.00142
TapTransformer	0.00085
XGBoost	0.00852

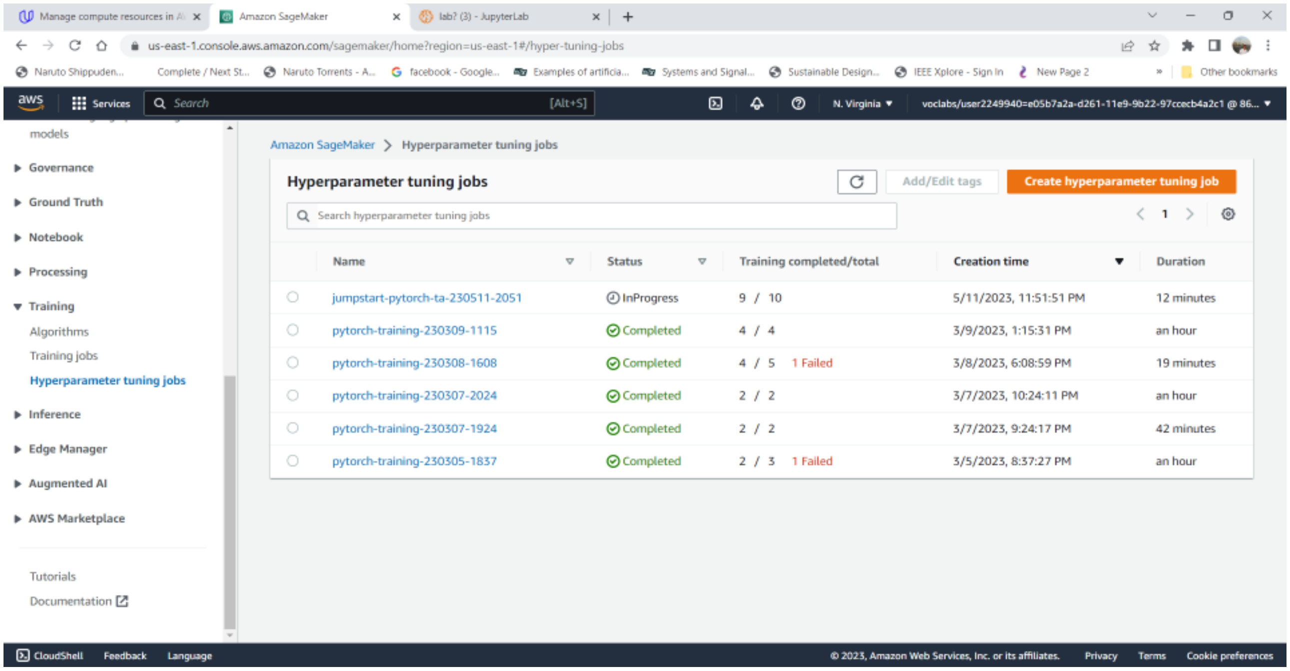
The following are figures to show the time series forecasting for XGBoost Regressor with the following hyperparameters:

XGBRegressor
(base_score=0.5, booster='gbtree', callbacks=None,
colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
early_stopping_rounds=50, enable_categorical=False,
eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
importance_type=None, interaction_constraints='',
learning_rate=0.01, max_bin=256, max_cat_to_onehot=4,
max_delta_step=0, max_depth=3, max_leaves=0, min_child_weight=1,
missing=nan, monotone_constraints='()', n_estimators=1000,
n_jobs=0, num_parallel_tree=1, objective='reg:linear',
predictor='auto', random_state=0, reg_alpha=0, ...)

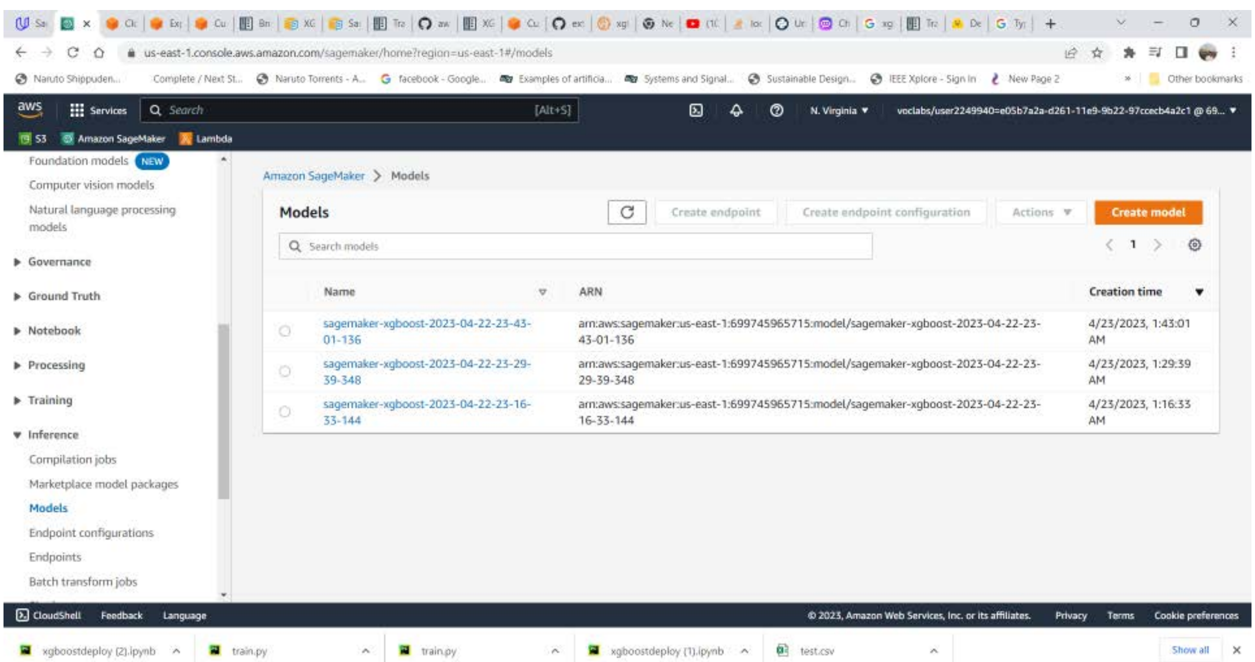


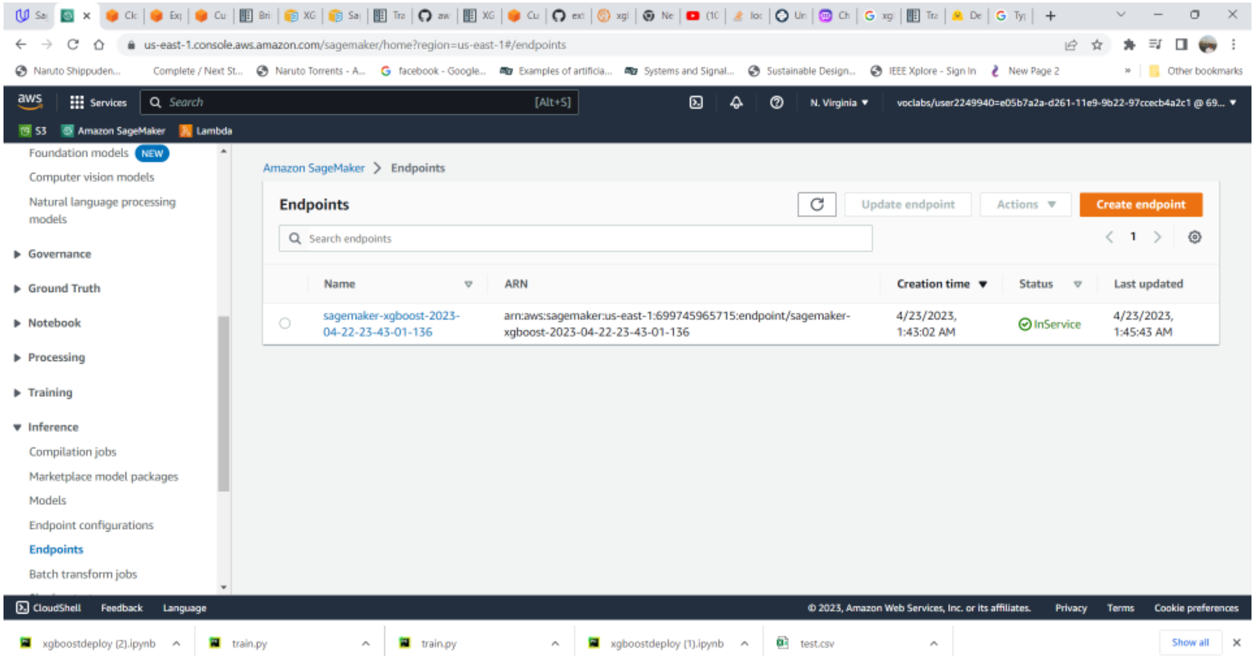


After we tested the above techniques (notebooks an
XGboost regressor with hyperparameter tunning

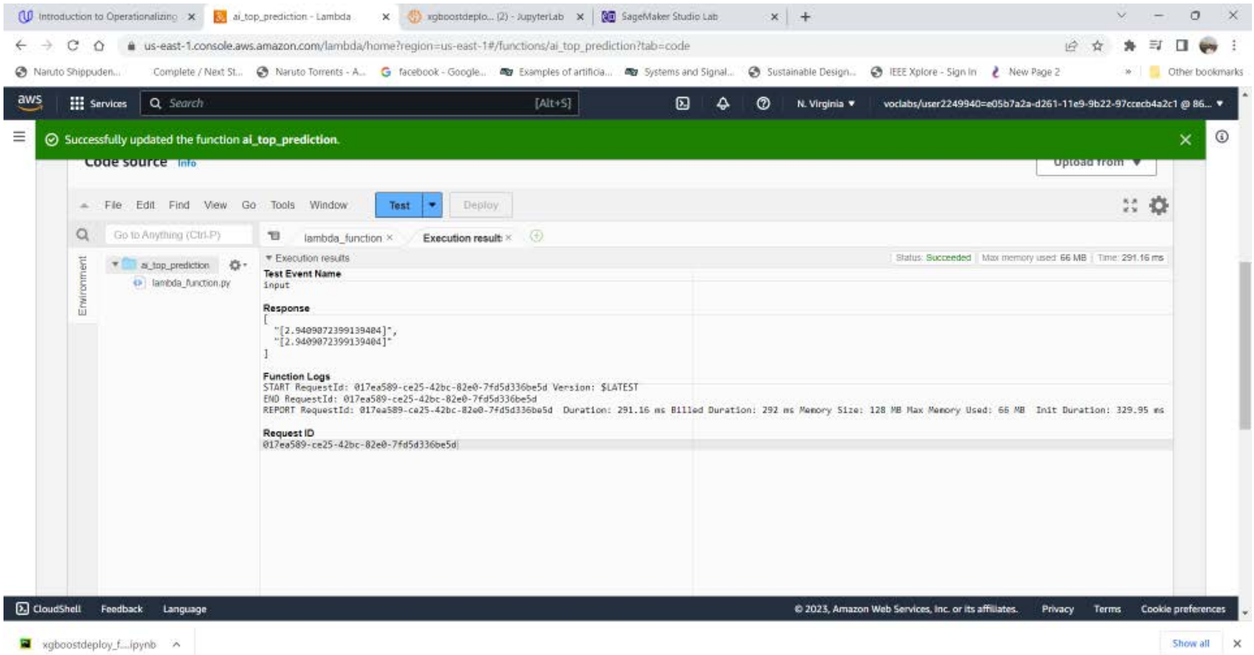


Then the model is deployed, and endpoint is created:

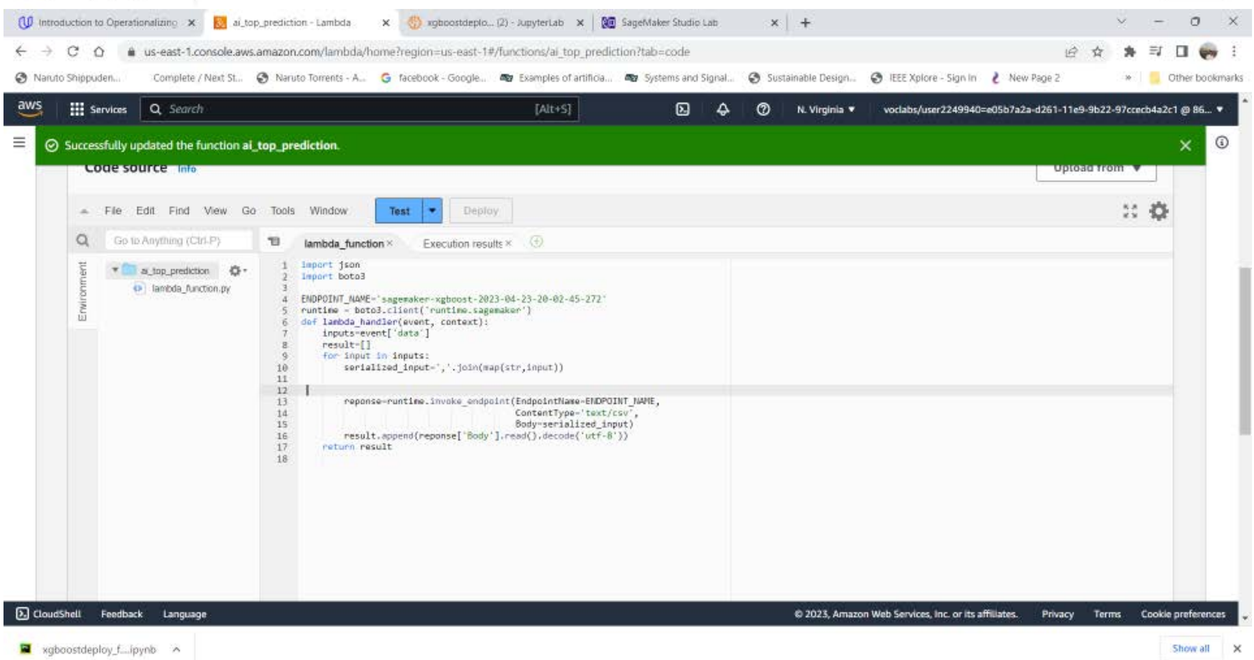




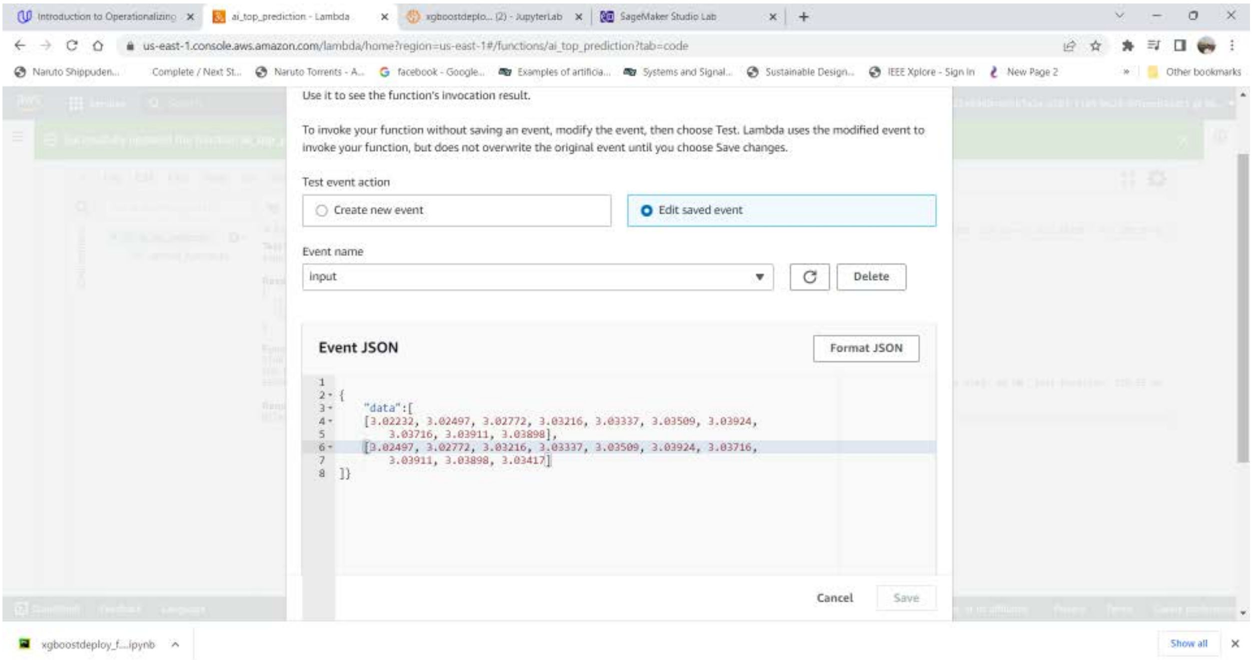
Then lambda function is created to envoke the end point:



Successfully updated the function ai_top_prediction.
Code source Info
Test
Deploy
lambda_function
Execution result
Execution results
Status: Succeeded
Max memory used: 66 MB
Time: 291.16 ms
Test Event Name
input
Response
[
"[2.9409072399139404]",
"[2.9409072399139404]"
]
Function Logs
START RequestId: 017ea589-ce25-42bc-82e0-7fd5d336be5d Version: $LATEST
END RequestId: 017ea589-ce25-42bc-82e0-7fd5d336be5d
REPORT RequestId: 017ea589-ce25-42bc-82e0-7fd5d336be5d Duration: 291.16 ms Billed Duration: 292 ms Memory Size: 128 MB Max Memory Used: 66 MB Init Duration: 329.95 ms
Request ID
017ea589-ce25-42bc-82e0-7fd5d336be5d

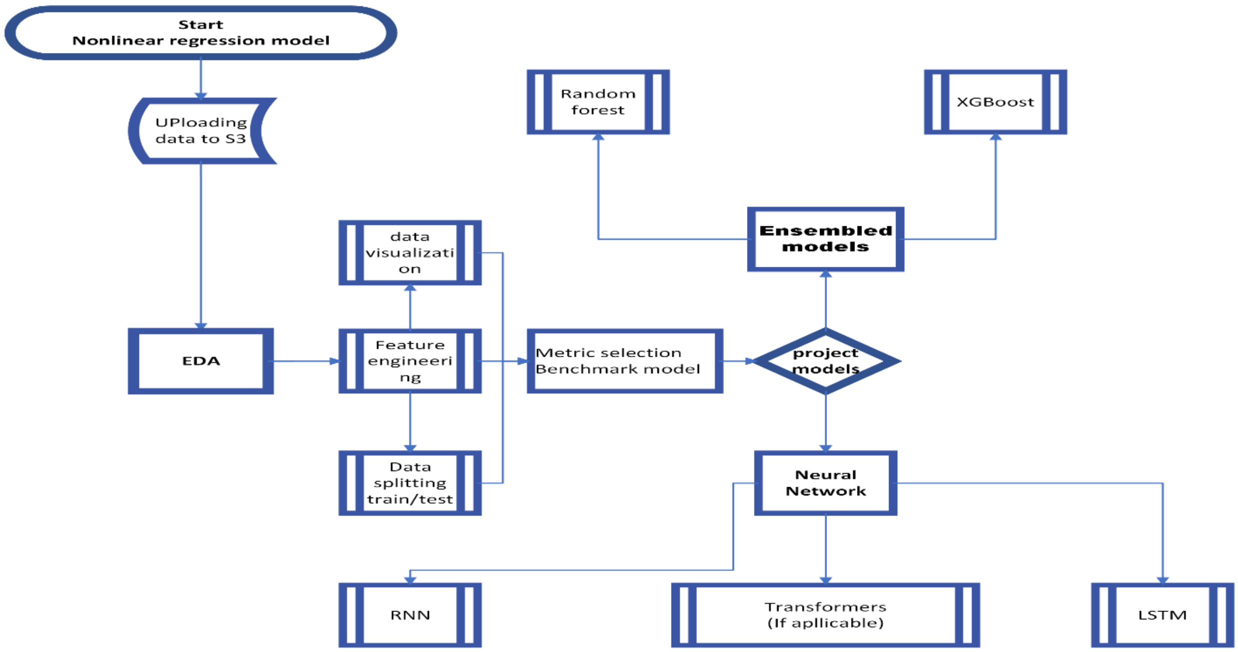


Successfully updated the function ai_top_prediction.
Code source Info
Test
Deploy
lambda_function
Execution results
import json
import boto3
ENDPOINT_NAME='sagemaker-xgboost-2023-04-23-20-02-45-272'
runtime = boto3.client('runtime.sagemaker')
def lambda_handler(event, context):
inputs=event['data']
result=[]
for input in inputs:
serialized_input=','.join(map(str,input))
reponse=runtime.invoke_endpoint(EndpointName=ENDPOINT_NAME,
ContentType='text/csv',
Body=serialized_input)
result.append(reponse['Body'].read().decode('utf-8'))
return result



Results, Observations, Conclusions:
The present study aimed to assess the efficacy of various machine learning algorithms in constructing a model suitable for employment in model predictive control (MPC). The findings of our study indicate that the TabTransformer neural network and XGBOOST exhibited notable performance and demonstrated significant promise for implementation in MPC. The TabTransformer neural network was observed to effectively model the intricate dynamics of the system and provide precise prognostications regarding the system's future behaviour. The implementation of this enhancement resulted in an enhancement of the operational efficiency of our Model Predictive Control (MPC) system, as it facilitated the system's ability to make judicious control determinations. Although LSTM showed a great potential in predicting the future system behavior it was prone to overfitting on many occasions.

Novel/Additive Information: The study's findings and observations suggest that the incorporation of machine learning, particularly the TabTransformer neural network, and XGBOOST into MPC has the capacity to notably enhance control performance. The results of our study indicate that additional investigation into the application of machine learning in model predictive control (MPC) is justified and has the potential to facilitate the creation of more sophisticated control systems.



Improvement:

The system should be optimized to cost in respect to value obtained from the gain of improving process variables.

1. Cutler and Ramaker. 1980_Cutler-Ramaker_Dynamic-matrix-control_JACC1980. 1980.
2. Richalet J, Rault A, Testud JL, Papon J. Model predictive heuristic control. Applications to industrial processes. Automatica. 1978;14(5):413–28.
3. Schwenzer M, Ay M, Bergs T, Abel D. Review on model predictive control: an engineering perspective. Int J Adv Manuf Technol. 2021;117(5–6):1327–49.
4. Bemporad A. Machine Learning Methods for Model Predictive Control. Slide. 2021;
5. Jain A, Morari M, Pappas GJ. Methods for Data-driven Model Predictive Control. ProQuest Diss Theses [Internet]. 2020;104. Available from: https://www.proquest.com/dissertations-theses/methods-data-driven-model-predictive-control/docview/2446708156/se-2?accountid=15179%0Ahttps://media.proquest.com/media/hms/PFT/2/jhrDH?_a=ChgyMDIyMTIxOTA3MDU1MDY4MDo1NDI5MDASBTk0MzI4GgpPTkVfU0VBUkNIIg4xNjUuMT