CHIYODA CORPORATION

CPW CHIYODA & PUBLIC WORKS Co., Ltd.

Specification of
AI Model Improvement Study for
LNG Plant AI Optimizer

1. Overview

1.1 Background and Objective

- In 2019 Chiyoda succeeded in developing the "LNG Plant AI Optimizer^{TM}".
  For the PT Donggi-Senoro LNG (DSLNG) plant in Indonesia Chiyoda confirmed increased LNG production through improved efficiency using the "LNG Plant AI Optimizer^{TM}".
- However, the optimal condition search process takes time, and it is necessary to prepare an environment for calculation for that process.
- In order to solve this problem, CYD and CPW developed the regression model which calculate the optimum operating parameters directly in the past fiscal year (FY2020).

- In the above process, DSLNG has been selected as a specific target LNG plant.
  However, considering product deployment for other LNG plants, it is desirable to develop a “General Model” which is not limited to a specific plant.
- In FY2020, CYD prepare the wider dataset which covers major LNG plants and develop the regression models for optimum operating parameters. However, the regression model accuracies are lower than specific plant case (DSLNG).
- In this study, hyper parameter tuning, optimization of pretreatment process, optimization of best parameter search algorism, etc. will be carried out to improve the model accuracies.

1.2 Study Policy

- The purpose of this study is to maximize the regression model accuracies.
- The scope is Natural Gas Liquefaction and Refrigeration System of LNG Plant.
- The original dataset is from traced simulation with wider range which covers major plants.

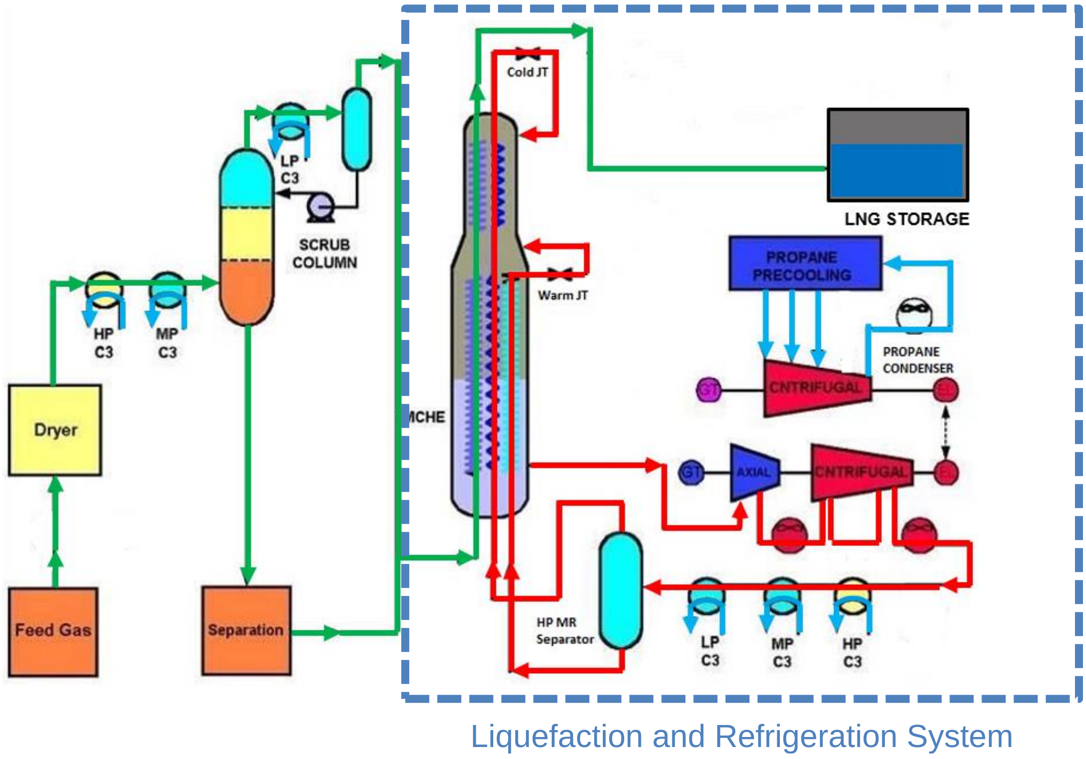
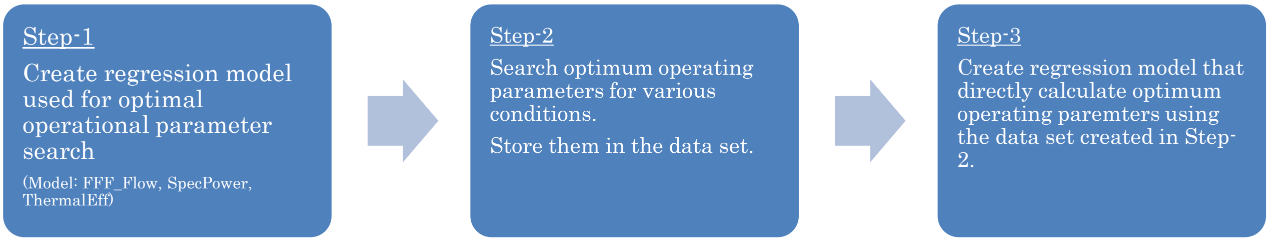


Fig. 1-1: Natural Gas Liquefaction Process

1.3 Overall Development Flow

- ◆ The overall work flow is similar with previous optimum parameter search study.
  The differences are:
  - The number of the dataset has been increased.
  - Increased Scope (Step-0: Optimization of Pretreatments)
- ◆ Overall flow to create the final regression model is shown below:
  As Step-0, optimization of pretreatments should be carried out.



Step-1
Create regression model used for optimal operational parameter search
(Model: FFF_Flow, SpecPower, ThermalEff)

Step-2
Search optimum operating parameters for various conditions.
Store them in the data set.

Step-3
Create regression model that directly calculate optimum operating paremters using the data set created in Step-2.

1.4 Communication

- ◆ Github Repository:
  https://github.com/efexis-com/LngGeneralModel_CPW
  (※ Keep large size files such as datasets in local environment, not on github.)
- ◆ Task communication:
  https://github.com/efexis-com/LngGeneralModel_CPW/issues

2. Details of Developments

2.1.1 Step-0 Optimization of dataset post-processing process

- ◆ Proceed post-processing using the preprocessed dataset provided by CGH:

  Dataset:
  https://github.com/efexis-com/LngGeneralModel_CPW/tree/master/output/preprocessed

  Post-Processing Code:
  https://github.com/efexis-com/LngGeneralModel_CPW/blob/master/visualize/data_post_processing.ipynb

- ◆ Study Cases:
  Since the extraction range of UA (of Main Cryogenic Heat Exchanger) affects the model accuracy. Please compare the following pre-treatment cases.

  Case A (Current Code):

  df_raw = df_raw[(df_raw['WB_UA'] > 1.5 * 10**7) & (df_raw['WB_UA'] < 6 * 10**7)]

  df_raw = df_raw[(df_raw['CB_UA'] > 0.75 * 10**6) & (df_raw['CB_UA'] < 3 * 10**6)]

Case B :

```
df_raw = df_raw[(df_raw['WB_UA'] > 1.5 * 10**7) & (df_raw['WB_UA'] < 8 * 10**7)]
df_raw = df_raw[(df_raw['CB_UA'] > 0.75 * 10**6) & (df_raw['CB_UA'] < 4 * 10**6)]
```

Case C:

```
df_raw = df_raw[(df_raw['WB_UA'] > 1.5 * 10**7) & (df_raw['WB_UA'] < 10 * 10**7)]
df_raw = df_raw[(df_raw['CB_UA'] > 0.75 * 10**6) & (df_raw['CB_UA'] < 5 * 10**6)]
```

Case Z:

No further extraction

◆ Evaluation:
Compare the accuracy of Regression of Thermal Efficiency as benchmark.

https://github.com/efexis-com/LngGeneralModel_CPW/blob/master/regression/thermaleff/thermaleff_gengt_1.ipynb

Note:

- Firstly, please carry out hyper parameter tuning with Case A post-processing in order to fix the hyper parameters for evaluation.
  (Step-0 Preparation Work)
- Since the number of the data is increased compared to previous data (dslng), epoch size can be reduced (to 200 or 500).

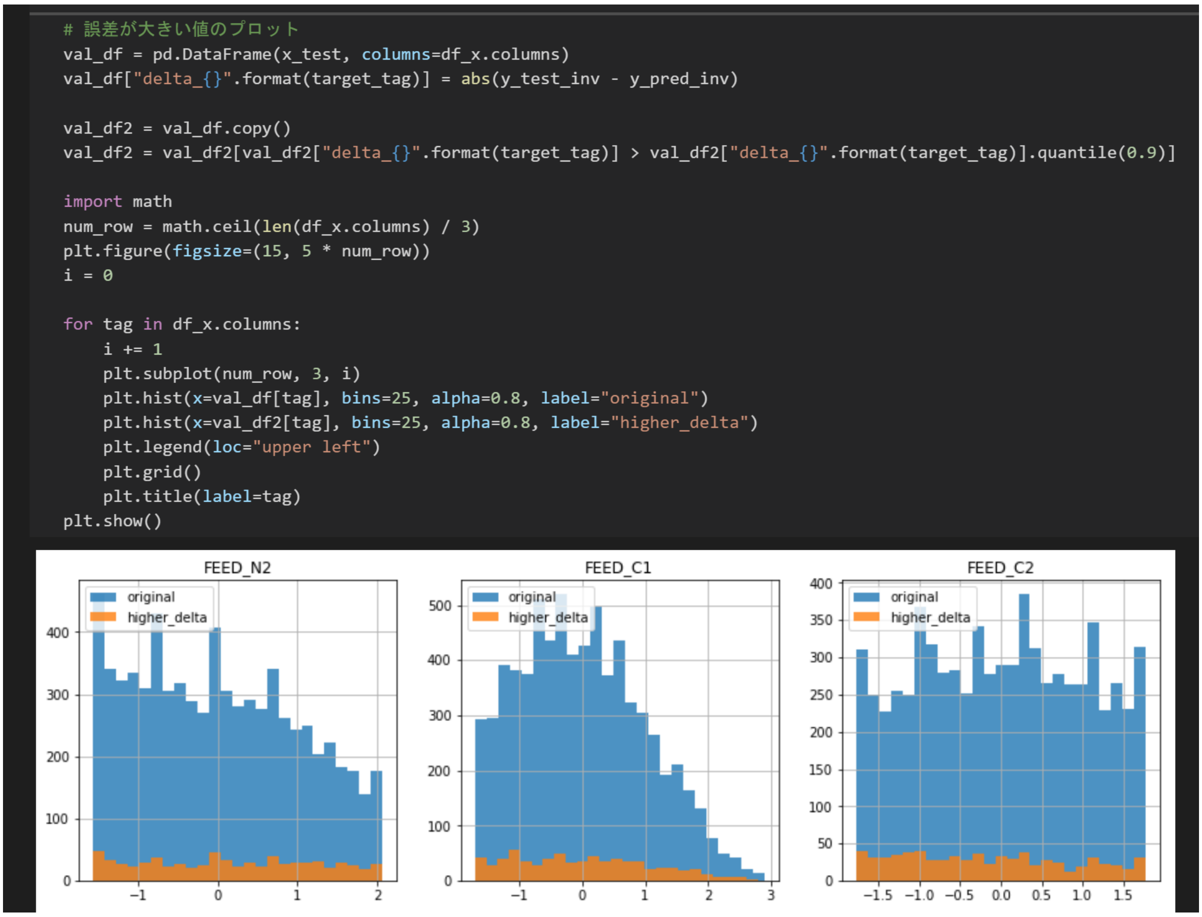
2.2 Step-1 Optimization of Regression Models for Optimum parameter Search

2.2.1 Overall Description

- Using the dataset created from post-processing process, proceed optimization of regression models required for the step-2.
  - Regression Models: FFF_Flow, SpecPower, ThermalEff, mche_ua, mche_out_temp

2.2.2 Coding

- Refer to the Original Source Codes:
  https://github.com/efexis-com/LngGeneralModel_CPW/tree/master/regression
- If the accuracy of the regression model is not good, consider excluding some part of the data based on the visualization results. (Fig. 2-1)
- Hyper Parameter Tuning
  - Proceed hyper parameter tuning same as Step-0 Preparation.



```
# 誤差が大きい値のプロット
val_df = pd.DataFrame(x_test, columns=df_x.columns)
val_df["delta_{}".format(target_tag)] = abs(y_test_inv - y_pred_inv)

val_df2 = val_df.copy()
val_df2 = val_df2[val_df2["delta_{}".format(target_tag)] > val_df2["delta_{}".format(target_tag)].quantile(0.9)]

import math
num_row = math.ceil(len(df_x.columns) / 3)
plt.figure(figsize=(15, 5 * num_row))
i = 0

for tag in df_x.columns:
    i += 1
    plt.subplot(num_row, 3, i)
    plt.hist(x=val_df[tag], bins=25, alpha=0.8, label="original")
    plt.hist(x=val_df2[tag], bins=25, alpha=0.8, label="higher_delta")
    plt.legend(loc="upper left")
    plt.grid()
    plt.title(label=tag)
plt.show()
```

Fig. 2-1: Data Distribution Comparison between original data and higher delta data

2.2.3 <u>Output</u>

- Regression Models (with optimized hyper paramters) and Scaler Files:
  FFF_Flow, SpecPower, ThermalEff, MCHE_OUT_T, MCHE_UA
- Folder:

  For Model and Scaler:

  https://github.com/efexis-com/LngGeneralModel_CPW/tree/master/AImodel

  For: Hyper parameter tuning:

  https://github.com/efexis-com/LngGeneralModel_CPW/tree/master/regression

2.3 Step-2 Optimization of optimum operating parameter search

2.3.1 Overall Description

- Proceed optimization of following optimum operating parameter search process.
- Please carry out the case study of optimum parameter search process.

Optimum operating parameter search process:

- Search optimum operating parameters for various conditions
  and Store them in the data set.
- Optimum operable parameters (list_v) are searched based on the external condition parameters (list_u) in the dataset (conditions.csv). The thermal efficiency (ThermalEff) and fuel from feed flow (FFF_Flow) of each combination of various operable parameters is calculated via regression models, and select the best combination of operable parameters based on the regression results (Fig. 2-2). The criteria for the selection is described in section 2.2.2.

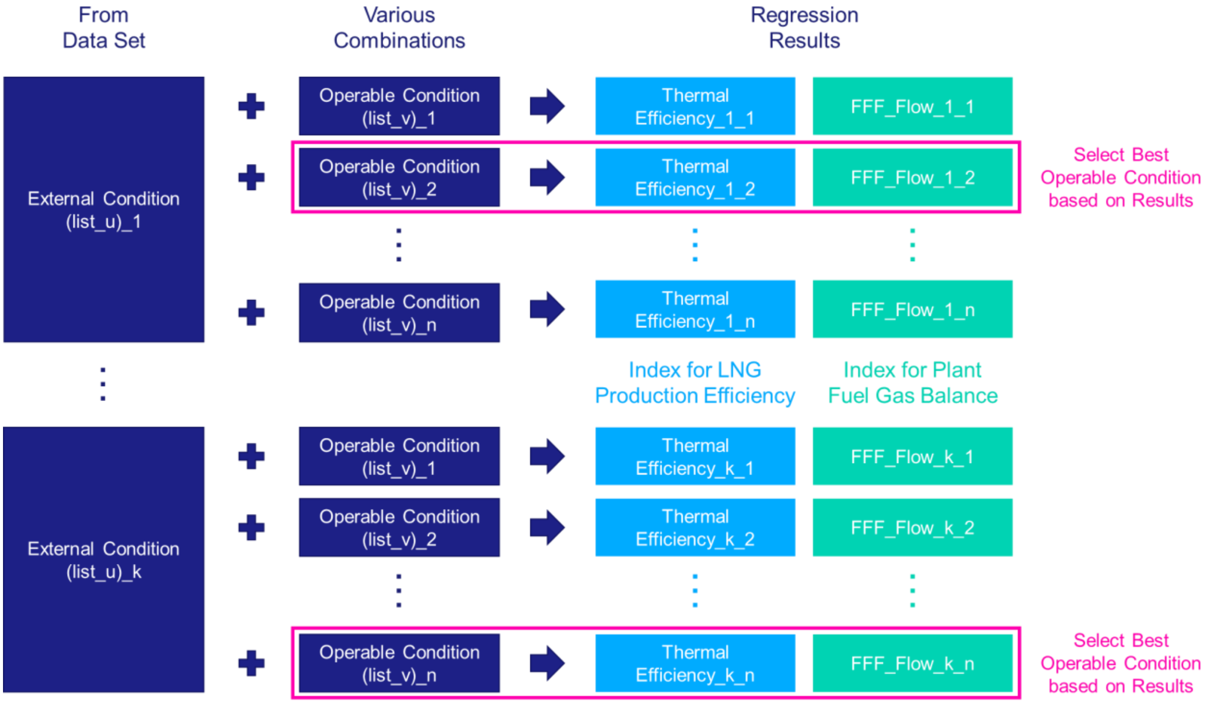


Fig. 2-2: Image of Optimum (Best) Parameter Search

2.3.2 Coding

- Refer to the Original Source Codes:

  https://github.com/efexis-com/LngGeneralModel_CPW/tree/master/best_search

- Model and Scaler:

  - Read Model and Scaler Files saved in Step1 folder.

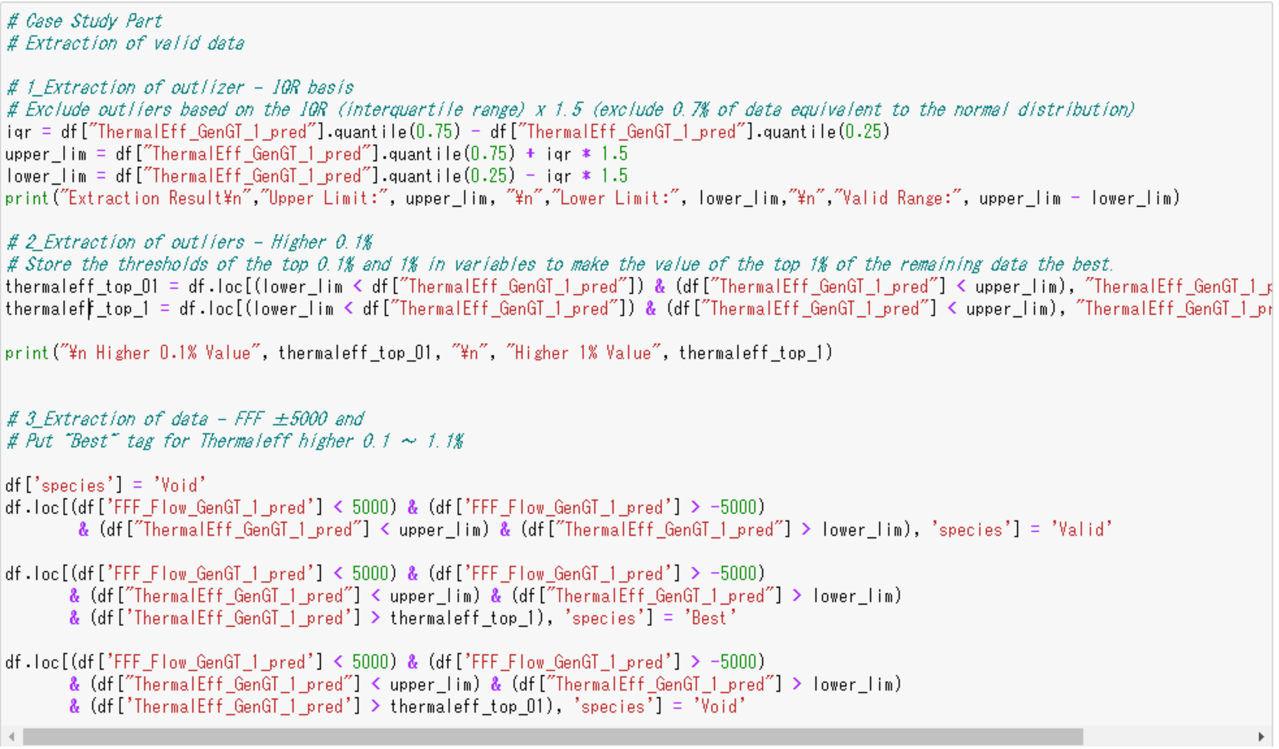
- Study Cases:

  Since an error occurs in the regression model, the following three processes such as exclusion of outliers are performed to stabilize the optimum value search. (Fig. 2-3)

  For these three processes, conduct a case study to improve the accuracy of the optimum value search process.

  Design the effective data extraction processes of the optimal value search by referring to the visualization by Validation Code, and compare/evaluate the results in the subsequent regression model creation (e.g. best thermal efficiency regression).
  In this process, as in Step-0, first, the subsequent Step-3 is executed under the fixed optimum value search condition to fix the hyper parameter for evaluation. (Step-2 Preparation Work).

  Validation Code:
  (https://github.com/efexis-com/LngGeneralModel_CPW/blob/master/best_search/best_parameter_search_validation_gengt.ipynb)



```
# Case Study Part
# Extraction of valid data

# 1_Extraction of outlizer - IQR basis
# Exclude outliers based on the IQR (interquartile range) x 1.5 (exclude 0.7% of data equivalent to the normal distribution)
iqr = df["ThermalEff_GenGT_1_pred"].quantile(0.75) - df["ThermalEff_GenGT_1_pred"].quantile(0.25)
upper_lim = df["ThermalEff_GenGT_1_pred"].quantile(0.75) + iqr * 1.5
lower_lim = df["ThermalEff_GenGT_1_pred"].quantile(0.25) - iqr * 1.5
print("Extraction Result¥n","Upper Limit:", upper_lim, "¥n","Lower Limit:", lower_lim,"¥n","Valid Range:", upper_lim - lower_lim)

# 2_Extraction of outliers - Higher 0.1%
# Store the thresholds of the top 0.1% and 1% in variables to make the value of the top 1% of the remaining data the best.
thermaleff_top_01 = df.loc[(lower_lim < df["ThermalEff_GenGT_1_pred"]) & (df["ThermalEff_GenGT_1_pred"] < upper_lim), "ThermalEff_GenGT_1_p
thermaleff_top_1 = df.loc[(lower_lim < df["ThermalEff_GenGT_1_pred"]) & (df["ThermalEff_GenGT_1_pred"] < upper_lim), "ThermalEff_GenGT_1_pr

print("¥n Higher 0.1% Value", thermaleff_top_01, "¥n", "Higher 1% Value", thermaleff_top_1)


# 3_Extraction of data - FFF ±5000 and
# Put "Best" tag for Thermaleff higher 0.1 ~ 1.1%

df['species'] = 'Void'
df.loc[(df['FFF_Flow_GenGT_1_pred'] < 5000) & (df['FFF_Flow_GenGT_1_pred'] > -5000)
        & (df["ThermalEff_GenGT_1_pred"] < upper_lim) & (df["ThermalEff_GenGT_1_pred"] > lower_lim), 'species'] = 'Valid'

df.loc[(df['FFF_Flow_GenGT_1_pred'] < 5000) & (df['FFF_Flow_GenGT_1_pred'] > -5000)
       & (df["ThermalEff_GenGT_1_pred"] < upper_lim) & (df["ThermalEff_GenGT_1_pred"] > lower_lim)
       & (df['ThermalEff_GenGT_1_pred'] > thermaleff_top_1), 'species'] = 'Best'

df.loc[(df['FFF_Flow_GenGT_1_pred'] < 5000) & (df['FFF_Flow_GenGT_1_pred'] > -5000)
       & (df["ThermalEff_GenGT_1_pred"] < upper_lim) & (df["ThermalEff_GenGT_1_pred"] > lower_lim)
       & (df['ThermalEff_GenGT_1_pred'] > thermaleff_top_01), 'species'] = 'Void'
```

Table 2-3: Valid data extraction during optimum parameter search

2.3.3 Output

- Data set including optimum (best) operable values.
  (Save your output in your local directly.)

2.4 Step-3 Optimization of optimum parameter regression models

2.4.1 Overall Description

- Using the dataset created in Step-2, optimize regression models of best parameters (mean values).

  - Data Set Tags:
    Original Datasets (FEED_N2, FEED_C1, FEED_C2, FEED_C3, FEED_iC4, FEED_nC4, FEED_C5+, AmbTemp, C3_CON_OUT, P_HPMR, P_LPMR, MCHE_BTM_DT, WB_UA, CB_UA, MR_N2, MR_C1, MR_C2, MR_C3, LMR_GASFLOW, FFF_Flow, SpecPower, ThermalEff, MCHE_OUT_T, MCHE_OUT_C_T, MCHE_MID_T, MCHE_MID_C_T, MCHE_IN_T, MCHE_IN_C_T, MR_GASFLOW, HMR_VFLOW, C3_GT_Power, LPMR_GT_Power, HPMR_GT_Power, LNG_Dens, Feasible, Note)
    + Best Parameters

  - Regression Models (Target Value):
    Best Parameters (e.g. Best_MR_N2, Best_ThermalEff)

2.4.2 Coding

- Same process as Step-1 (Target Values are only changed)
  Reference: https://github.com/efexis-com/LngGeneralModel_CPW/tree/master/regression/best

3. Overall Output

3.1 Deliverables

- Program (or Source Code)
  - Python (3.8), Tensor Flow (2.3) with Keras

- Description of “how to use the program, how to install it, the libraries you use, etc. (You may include comments in the source code.)”
  ※ Not Required in case of using the typical libraries in the program.

- Report (Includes empirical results, a brief description of the algorithm used, and problems)

  Reference for execution summary:
  https://github.com/efexis-com/LngGeneralModel_CPW/tree/master/summary

3.2 Scheduled Date for Deliverables

- End of July, 2021 (As of 21st of June)