Final: PM2.5 Prediction with Machine Learning*

*For the fulfillment project proposal of AT82.01 Computer Programming for Data Science and Artificial Intelligence course by Dr. Chantri Polprasert

1st Kyi Thin Nu (group 7)
Department of Data Sciences and Artificial Intelligence
Asian Institute of Technology
Pathum Thani, Thailand
st124087@ait.asia

2nd Thongtong Eamsaard (group 7)
Department of Industrial System Engineering
Asian Institute of Technology
Pathum Thani, Thailand
st123300@ait.asia

***Abstract*—The project is to forecast PM2.5 based on weather data set collected from five weather stations in Bangkok. Using the scikit-learn library in Python, we created the Random Forest prediction model that can predict PM2.5 given the weather data. Then, we deployed the model into website created by Flask, to AWS for end-users to use.**

***Index Terms*—Particulate Matters, PM2.5, Machine Learning**

I. Introduction

A. Introduction to PM2.5

PM2.5, or Particulate Matter 2.5, is a critical measure of air quality that refers to tiny airborne particles or droplets with a diameter of 2.5 micrometers or smaller. These minuscule particles can originate from a variety of sources, including industrial emissions, vehicle exhaust, construction activities, natural dust, and even chemical reactions in the atmosphere. PM2.5 is significant because it has a substantial impact on both human health and the environment. Understanding PM2.5 levels is essential for assessing air quality, making informed policy decisions, and implementing measures to safeguard public well-being.

Fig. 1: PM 2.5 Impact on environment and humans [8]

B. Global Perspective View on PM2.5

A global perspective on PM2.5 levels is vital to comprehend the scale and variations in air quality across different regions. By monitoring PM2.5 on a global scale, we can identify trends, sources of pollution, and areas where air quality may be particularly hazardous. This global view often involves the use of satellites and international air quality monitoring networks. It helps nations collaborate in addressing transboundary air pollution and sharing information to mitigate the impact of airborne particles on a global scale.

C. Local Perspective View for Thailand

On a local level, such as within a country like Thailand, monitoring PM2.5 is crucial for assessing the immediate air quality conditions that people are exposed to. Local monitoring networks, government agencies, and environmental organizations collect data on PM2.5 levels to provide real-time information to citizens. This local perspective helps individuals make informed decisions about outdoor activities, and it assists policymakers in implementing measures to improve air quality and protect public health.

Fig. 2: PM 2.5 Impact in Thailand [9]

D. Why Do We Do PM2.5 Projects?

1) Protecting Public Health: PM2.5 particles are so small that they can penetrate deep into the respiratory system, posing significant health risks. PM2.5 projects aim to reduce exposure to these particles and thereby protect the health of communities. High PM2.5 levels have been linked to respiratory diseases, cardiovascular issues, and even premature death.

2) Environmental Impact: PM2.5 particles can also harm the environment. They can contribute to smog formation, damage ecosystems, and affect water quality. PM2.5 projects seek to reduce these environmental impacts.
3) Policy and Regulation: Monitoring PM2.5 is essential for setting air quality standards and regulations. Governments and regulatory agencies use PM2.5 data to implement measures to limit emissions from various sources.
4) Awareness and Education: PM2.5 projects help raise public awareness about air quality issues. They encourage people to take action to reduce their own contributions to air pollution and to advocate for cleaner air.

In summary, PM2.5 projects serve the vital purpose of safeguarding both human health and the environment. They provide essential information for informed decision-making, regulation, and action at both the local and global levels. The business understanding are in Fig. 3.

II. Problem Statement

To predict the PM2.5 values based on given weather conditions and trends of the PM2.5, Temporal Trends PM2.5 (seasonal, monthly, daily, hourly), Correlations between weather parameters and PM2.5, Weather effects (wind speed and temperature) on PM2.5

III. Related Works

This project goal is to train the model to able to predict the PM2.5. Here the focus is on how other concerned in PM2.5 at different area and weather, some well-known techniques to make the model be able to predict desired target variables. Thus, These can make the model predicted the PM2.5 with high accuracy, precision, and recall.

Vahid Mehdipour [4] and his team from Tehran compared different methods for modeling PM2.5 in the capital city of Iran, Tehran. They proposed decision trees (DT), Bayesian Network (BN), and support vector machine (SVM). Using the data for over three periods, they concluded that PM10, NO_2, SO_2, and O_3 are critical factors for PM2.5 with the best model is SVM.

Delhi, another mega-city in India, also faced an enormous of air pollution because of rapid development for a while. Nidhi Sharma and her colleagues [6] forecast pollution load in an atmosphere using time-series regression forecasting. In the results, predicted trends are shown after 2017.

Another interesting paper used Taiwan Air Quality Monitoring (TAQMN) data set. Doreswamy and his team [7] did the forecasting using also machine learning regression models. The data used are from 2012 to 2017. Models were evaluated by Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Squared Error (MSE), and Coefficient of Determination (R^2). They used Fourier arrangement and spline multinomial to fill the missing values in data set. The model they used are random forest regressor (RFR), gradient boosting regressor GBR), k neighbors regressor (KNR), MLP regressor (MLPR), and decision tree regressor CART. To select the best model, they used cross-validation and determined that gradient boosting regressor model is better in forecasting air pollution in TAQMN data.

IV. Dataset

A. Description

The data set contains a record of PM2.5 per hour recorded in 2019 from 5 stations distributed in Bangkok as shown in Fig. 7. In total, 5 stations' data set for this project are given by Dr. Chantri via the Pollution Control Department of Thailand. The station are numbered as follows (5 from all 66 stations established in 2019).

- Station 03: Bang Khun Thian, Bangkok
- Station 50: Pathum Wan, Bangkok
- Station 52: Thonburi, Bangkok
- Station 53: Chok Chai, Bangkok
- Station 54: Din Daeng, Bangkok

The data set is in EXCEL spread sheet format.

B. Features

The dataset contains following features:

- Date and time of record
- Various air quality parameters (CO, NO, NO_2, NOX, O_3, PM10, PM2.5)
- Meteorological data (wind speed, wind direction, temperature)

V. Methodology

To begin with, this project use scikit-learn [5] packages as it is simple and efficient tools for predictive data analysis. Next, we received data set, cleaned data, explored some data insights, pre-processed, selected the model, trained the model, evaluation, and deployed into simple website in AWS platform to demonstrate our powerful PM2.5 prediction.

A. Data Acquisition

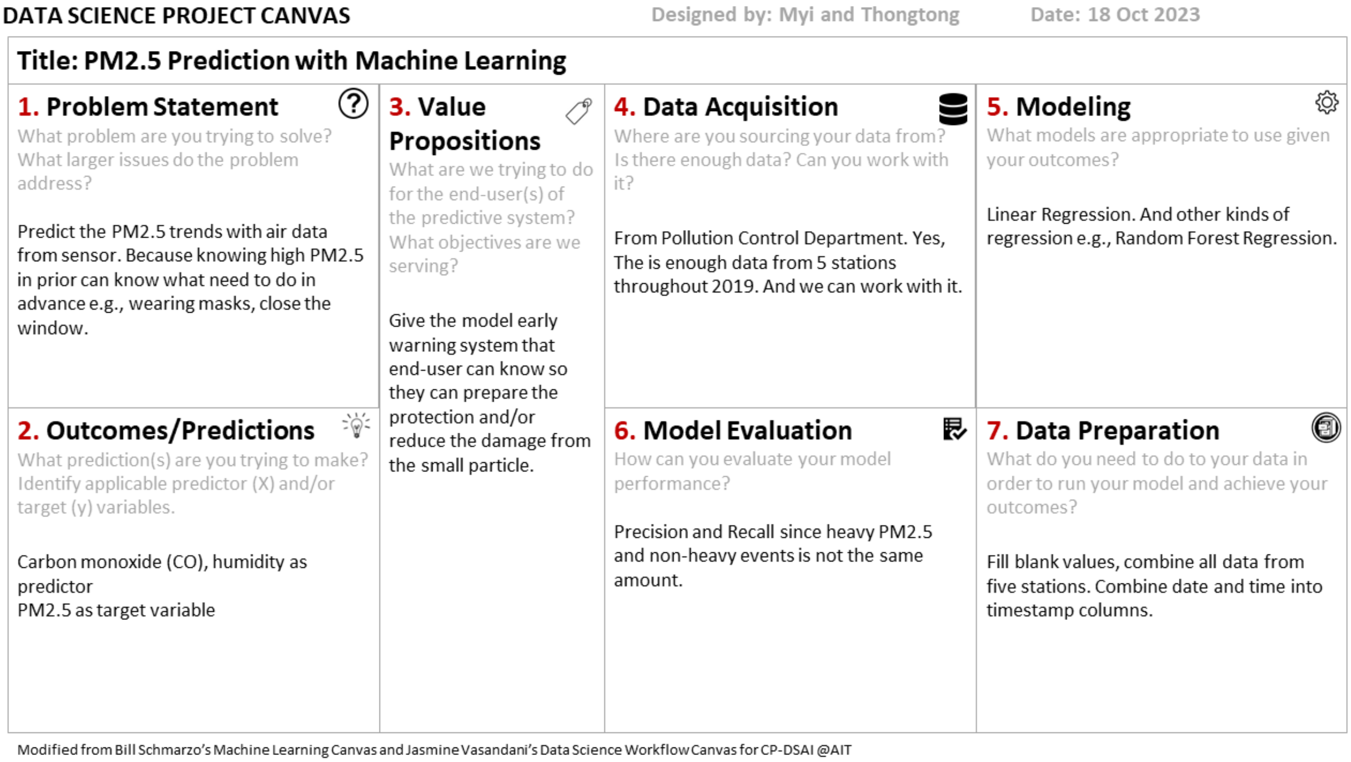
To obtain the data set, Professor Chantri received the data set for us. These data sets came from the Pollution Control Department of Thailand (PCD) [2]. Generally, the data are recorded daily for public use. However, we can ask PCD for more details hourly records. Original data looks as shown in Fig. 8.

The original filename are in Thai. For better communicate between groups, we agreed to change into more accessible file names in English as shown in Fig. 9. We then imported the data set from each station, and stored in variables `station03`, `station50`, `station52`, `station53`, and `station54`, respectively.

B. Clean Data

As the original data were not generalized well and were really messy, we needed some cleaning process before analyzed with Exploratory Data Analysis in V-C. The process included here are:

- Dropped second level headers
- Renamed column names in Fig. 10



DATA SCIENCE PROJECT CANVAS   Designed by: Myi and Thongtong   Date: 18 Oct 2023

Title: PM2.5 Prediction with Machine Learning

1. Problem Statement

What problem are you trying to solve? What larger issues do the problem address?

Predict the PM2.5 trends with air data from sensor. Because knowing high PM2.5 in prior can know what need to do in advance e.g., wearing masks, close the window.

2. Outcomes/Predictions

What prediction(s) are you trying to make? Identify applicable predictor (X) and/or target (y) variables.

Carbon monoxide (CO), humidity as predictor
PM2.5 as target variable

3. Value Propositions

What are we trying to do for the end-user(s) of the predictive system? What objectives are we serving?

Give the model early warning system that end-user can know so they can prepare the protection and/or reduce the damage from the small particle.

4. Data Acquisition

Where are you sourcing your data from? Is there enough data? Can you work with it?

From Pollution Control Department. Yes, The is enough data from 5 stations throughout 2019. And we can work with it.

5. Modeling

What models are appropriate to use given your outcomes?

Linear Regression. And other kinds of regression e.g., Random Forest Regression.

6. Model Evaluation

How can you evaluate your model performance?

Precision and Recall since heavy PM2.5 and non-heavy events is not the same amount.

7. Data Preparation

What do you need to do to your data in order to run your model and achieve your outcomes?

Fill blank values, combine all data from five stations. Combine date and time into timestamp columns.

Modified from Bill Schmarzo's Machine Learning Canvas and Jasmine Vasandani's Data Science Workflow Canvas for CP-DSAI @AIT

Fig. 3: Project Canvas

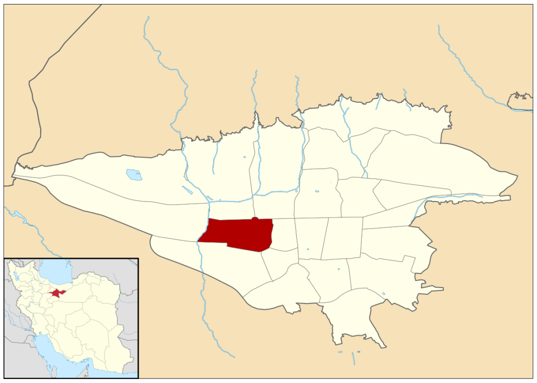


Fig. 4: Area of interest, Tehran, Iran [4]

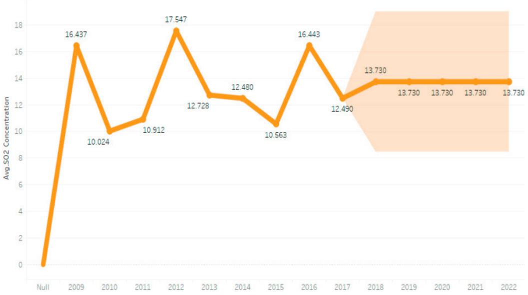


Fig. 5: trend of SO_2 in $\mu g/m^3$ [6]

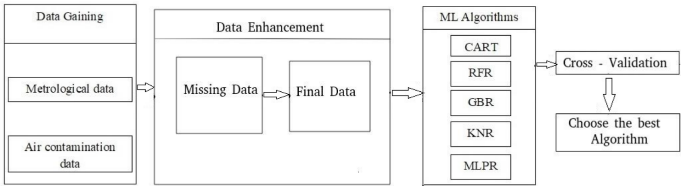


Fig. 6: Proposed prediction pipeline model for air pollution [7]

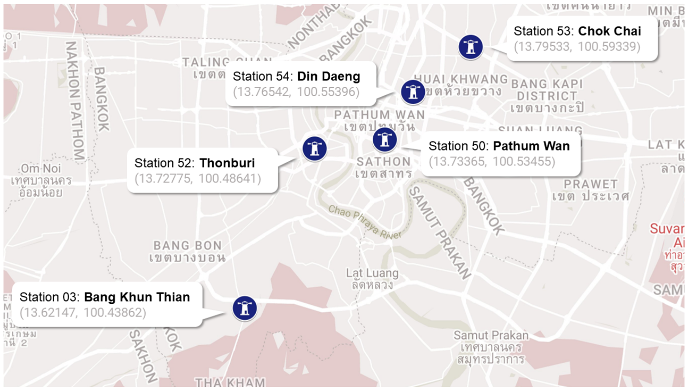
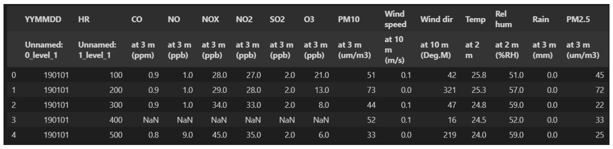


Fig. 7: Weather Stations Location



	YYMMDD	HR	CO	NO	NOX	NO2	SO2	O3	PM10	Wind speed	Wind dir	Temp	Rel hum	Rain	PM2.5
	Unnamed: 0_level_1	Unnamed: 1_level_1	at 3 m (ppm)	at 3 m (ppb)	at 3 m (ppb)	at 3 m (ppb)	at 3 m (ppb)	at 3 m (ppb)	at 3 m (um/m3)	at 10 m (m/s)	at 10 m (Deg.M)	at 2 m	at 2 m (%RH)	at 3 m (mm)	at 3 m (um/m3)
0	190101	100	0.9	1.0	28.0	27.0	2.0	21.0	51	0.1	42	25.8	51.0	0.0	45
1	190101	200	0.9	1.0	29.0	28.0	2.0	13.0	73	0.0	321	25.3	57.0	0.0	72
2	190101	300	0.9	1.0	34.0	33.0	2.0	8.0	44	0.1	47	24.8	59.0	0.0	22
3	190101	400	NaN	NaN	NaN	NaN	NaN	NaN	52	0.1	16	24.5	52.0	0.0	33
4	190101	500	0.8	9.0	45.0	35.0	2.0	6.0	33	0.0	219	24.0	59.0	0.0	25

Fig. 8: Original sensor data from Thonburi station, Bangkok

- Added Province and District columns
- Merged all fives stations into single `DataFrame`
- Merged Date and Time columns and converged into Timestamp (datetime formatting)
- Reordered the columns
- Checked and edited data types of columns (`dtypes`)
- Replaced string values in numerical columns

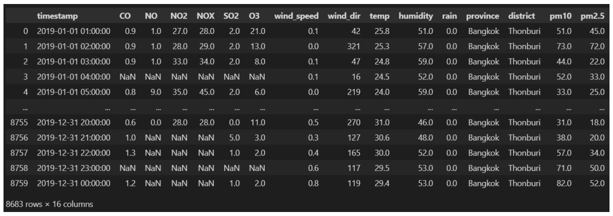
```
# rename columns
df_52.rename(columns = {’YYMMDD’:’yymmdd’,
                        ’HR’:’hr’,
                        ’CO’:’CO’,
                        ’ NO ’:’NO’,
                        ’ NOX ’:’NOX’,
                        ’ NO2’:’NO2’,
                        ’ SO2 ’: ’SO2’,
                        ’ Wind speed’: ’wind_speed’,
                        ’ Wind dir’: ’wind_dir’,
                        ’PM10’:’pm10’,
                        ’PM2.5’:’pm2.5’,
                        ’ Temp’:’temp’,
                        ’ Rel hum’:’humidity’,
                        ’ Rain’:’rain’
                         }, inplace = True)
```

Fig. 10: Renaming all column names

C. Exploratory Data Analysis (EDA)

For this task, we just exploring the data to see if there are potential problems in the data set (outliers, mislabeled data, unwanted correlations between variables/samples, etc.). However, no actual work done in here. The steps includes identifying missing values and deciding how to handle them, whether by imputing missing values, removing rows with missing values, or using other strategies.

Then, after we did some more cleansing and format the data, we could show some insights in the data as shown in Fig. 11.



	timestamp	CO	NO	NO2	NOX	SO2	O3	wind_speed	wind_dir	temp	humidity	rain	province	district	pm10	pm2.5
0	2019-01-01 01:00:00	0.9	1.0	27.0	28.0	2.0	21.0	0.1	42	25.8	51.0	0.0	Bangkok	Thonburi	51.0	45.0
1	2019-01-01 02:00:00	0.9	1.0	28.0	29.0	2.0	13.0	0.0	321	25.3	57.0	0.0	Bangkok	Thonburi	73.0	72.0
2	2019-01-01 03:00:00	0.9	1.0	33.0	34.0	2.0	8.0	0.1	47	24.8	59.0	0.0	Bangkok	Thonburi	44.0	22.0
3	2019-01-01 04:00:00	NaN	NaN	NaN	NaN	NaN	NaN	0.1	16	24.5	52.0	0.0	Bangkok	Thonburi	52.0	33.0
4	2019-01-01 05:00:00	0.8	9.0	35.0	45.0	2.0	6.0	0.0	219	24.0	59.0	0.0	Bangkok	Thonburi	33.0	25.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8755	2019-12-31 20:00:00	0.6	0.0	28.0	28.0	0.0	11.0	0.5	270	31.0	46.0	0.0	Bangkok	Thonburi	31.0	18.0
8756	2019-12-31 21:00:00	1.0	NaN	NaN	NaN	5.0	3.0	0.3	127	30.6	48.0	0.0	Bangkok	Thonburi	38.0	20.0
8757	2019-12-31 22:00:00	1.3	NaN	NaN	NaN	1.0	2.0	0.4	165	30.0	52.0	0.0	Bangkok	Thonburi	57.0	34.0
8758	2019-12-31 23:00:00	NaN	NaN	NaN	NaN	NaN	NaN	0.6	117	29.5	53.0	0.0	Bangkok	Thonburi	71.0	50.0
8759	2019-12-31 00:00:00	1.2	NaN	NaN	NaN	1.0	2.0	0.8	119	29.4	53.0	0.0	Bangkok	Thonburi	82.0	52.0

8683 rows × 16 columns

Fig. 11: data after do some cleansing before put into EDA

At first, We do uni-variate analysis, plotting to see the trend of average PM2.5 daily in all 5 stations in Fig. 12. PM2.5 is quite high in the first two months. Then it drops dramatically at each its lowest in September. Then, it starts rising again up until the end of December. Additionally, we also did some histogram plot to see the distribution of PM2.5 in Fig. 13. It normal range is from between 15 to 35 most of the times. Then, we did some scatter plot to see more insight between PM2.5 and PM10 of that predictor in Fig. 14. As predicted, it shows that PM2.5 strongly related to PM10. Furthermore, if these two are high, CO value will also be high too.

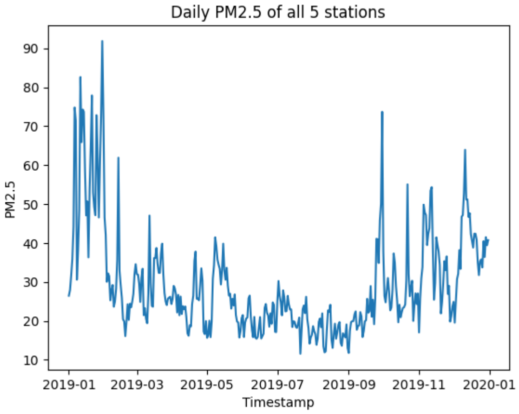


Fig. 12: Daily mean PM2.5 of all station

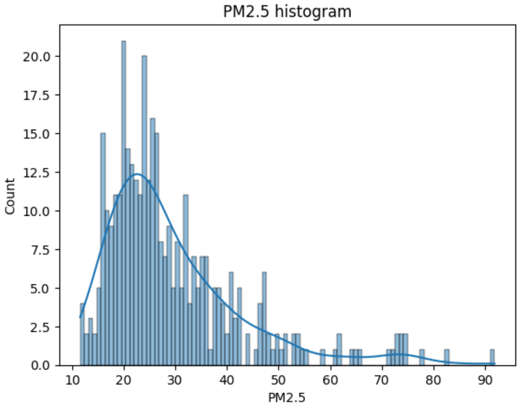
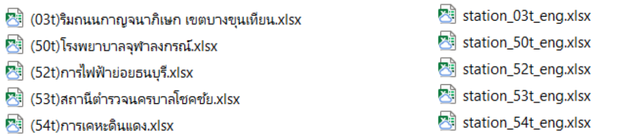


Fig. 13: PM2.5 histogram

Next, we compared PM2.5 from each station and do some kernel density estimate (kde) distribution in Fig. 15 and Fig. 16, respectively. Since all stations are relatively in the same province, Bangkok, PM2.5 values should be in the same trends most of the time. However, in kde plot for visualizing the distribution of observations in a dataset, analogous to a histogram, it shows that stations has 2 different distributions of the PM2.5 values. Showing that the kde distribution of Bang Khun Thian is similar with Din Daeng. The other group is Chok Chai, Pathum Wan, and Thonburi. Also, we do some scatter on PM2.5 for each station and it's quite messy in Fig. 17. In this figure, even though not clearly see at first, if NO_2 increases, it tends to increase the minimum PM2.5 value boundary. This indicates that NO_2 has some effects to PM2.5.



(03t)ริมถนนกาญจนาภิเษก เขตบางขุนเทียน.xlsx
(50t)โรงพยาบาลจุฬาลงกรณ์.xlsx
(52t)การไฟฟ้าย่อยธนบุรี.xlsx
(53t)สถานีตำรวจนครบาลโชคชัย.xlsx
(54t)การเคหะดินแดง.xlsx

(a) Original Filenames

station_03t_eng.xlsx
station_50t_eng.xlsx
station_52t_eng.xlsx
station_53t_eng.xlsx
station_54t_eng.xlsx

(b) Renamed Filenames

Fig. 9: Rename filenames from Thai to English for easy access

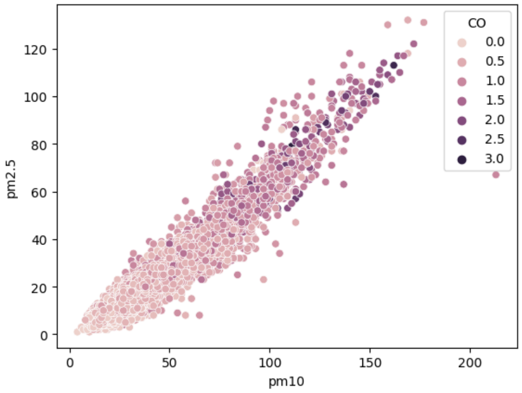


Fig. 14: Scatter plot of PM2.5-PM10 and colored by CO

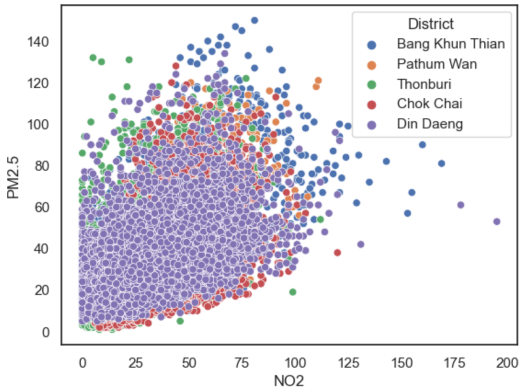


Fig. 17: Scatter plot of PM2.5-NO_2 and colored by District

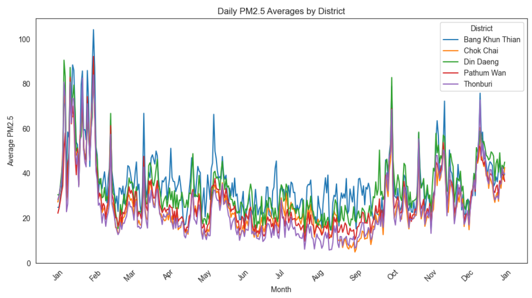


Fig. 15: PM2.5 for each stations

Lastly, We did multivariate analysis. We did the correlation heat map plot to see potential for predictor to use in Fig. 18a. It can be seen that CO, NO, NO_2, NOX, SO_2, and PM10 have potential to be the predictor for PM2.5 as target variable. We also computed the predictive power score (PPS) in Fig. 18b, which calculate how strong each predictor can predict other (target) variables. This time, PPS states that only NO_2 and PM10 has some predictive power for PM2.5. We created the violin plot to see the distribution, outliers, mean, and minimum and maximum range for each numerical variables as shown in Fig. 19. We see that most of all variables are not normal distribution, but we need to confirmed again with the probability plot before we decide to use what method for scaling. We also have some interesting test that, wind direction and wind speed is somewhat contribute to PM2.5. In additions, it can also show that most of the time, where wind are coming from which direction in Fig. 20.

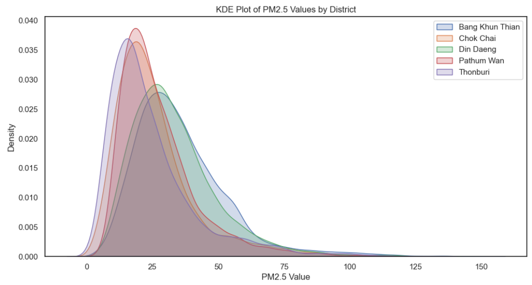
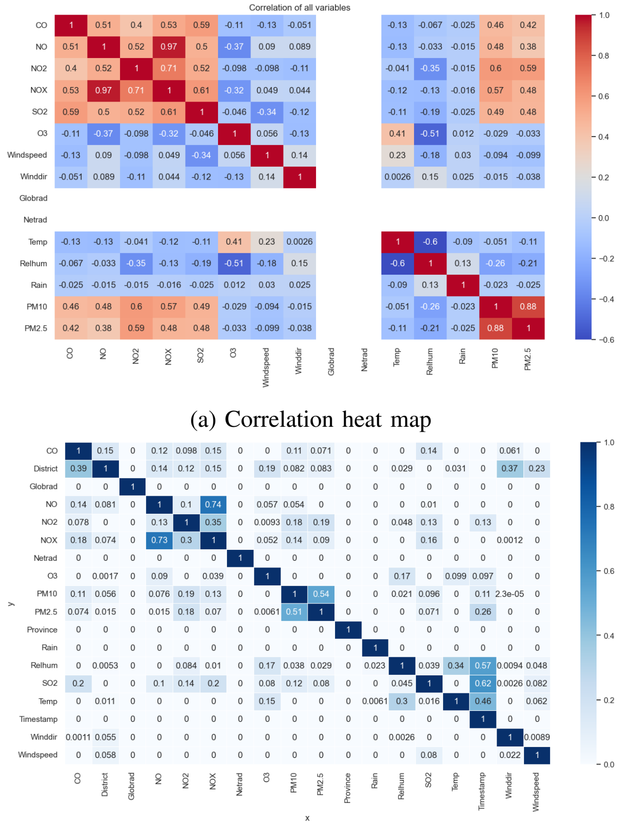


Fig. 16: kde distribution for each stations



(a) Correlation heat map

(b) Power score

Fig. 18: Correlation and Power score between variables

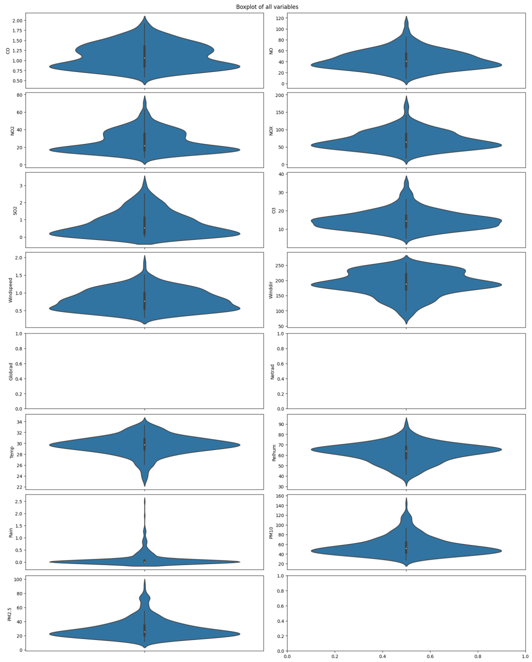


Fig. 19: Violin plot of all variables

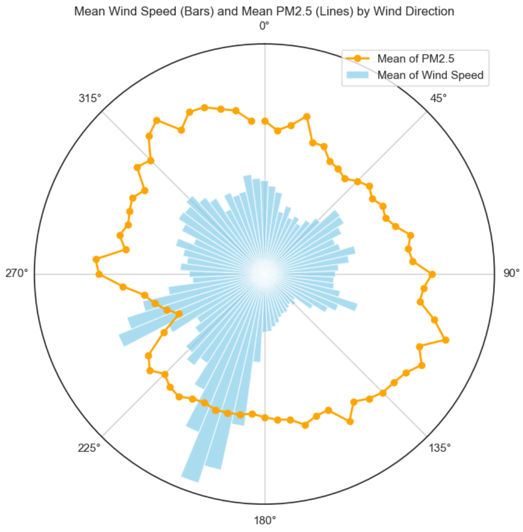


Fig. 20: Mean Wind Speed and Mean PM2.5 by Wind Direction

D. *Pre-processing*

After explored what our data set looks like, we did pre-processing - the steps required to go from raw data to a format suitable to input to your ML model [1].

1) Remove Unnecessary Columns: These columns will be removed:

- ‘Globrad’ and ‘Netrad’ will be removed because 100 % are NaN (Not a Number), they don’t have any reference numbers to fill.
- ‘SO2’ will also be removed because not enough data. Only less than 20 % entries are not enough to fill the rest more than 80 % of the missing values.
- ‘Province’ has only single unique value, Bangkok. No variation anyway.
- ‘Timestamp’ will also be dropped in this stage assuming time-independent.

2) Train-test Split: We split data set into train and test set with test ratio of 0.2 with fixed random state. In order to make train and test set has the same data from each station equally, we defined stratify by District column. We checked that train set are equally distributed on District in Fig 21. The train-test split need to be random (shuffled). If not, it’ll be ordered chronologically and station-orderly, making the last station has limited train data. The train test split settings is shown in Fig. 22. After that, we then imputed, encoded, and scaled the data.

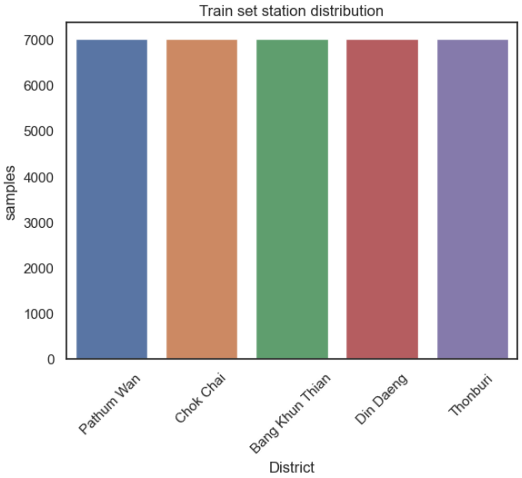


Fig. 21: Train set District (station) distribution check

```
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
                random_state=40,
                shuffle=True,
                stratify=X['District'])
```

Fig. 22: Train test split

3) Replace missing values (Imputation): Since all columns are received from sensor data. Assuming that all data were calibrated correctly, we filled with forward fill, and some for backward fill. The imputation process is shown in Fig. 23. The reason we did imputed step separately, not included in Pipeline is because in `SimpleImputer()`, there's no forward fill and backward fill option available. Due to shuffle option in V-D2, the index are now not in order because of randomly shuffled, and are not in chronological order any more. So, the forward fill will fill wrongly. Thus, After the `train_test_split`, do `.reindex` method to make it in order of station and chronologically again (not exact) but it stills better, after that, do forward fill and backward fill.

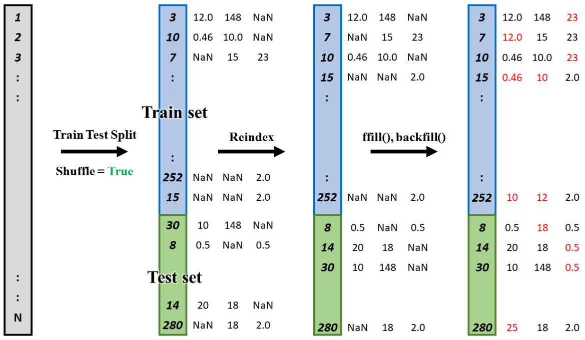


Fig. 23: Imputation process

4) Scaling: Every numerical columns come at different ranges in Fig. 24. So we need to do the scaling, otherwise the model may neglect some of the predictor(s) and pass some of them.

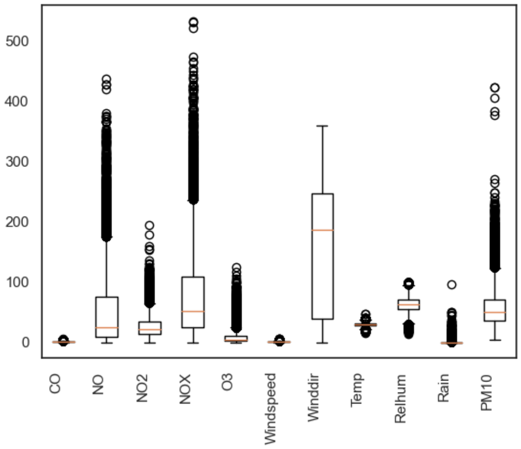


Fig. 24: Predictor variables at different scales

In order to select the appropriate scaler type, we need to check at probability plot of each predictor separately. If it was a normal distribution, we'd scale it with StandardScaler(). If it was not, we'll scale it with MinMaxScaler(). Luckily, we can check the normal distribution using a probability plot from the stats library in Fig. 25. In this case, NO_2 should use MinMaxScaler() because this indicates NO_2 is not in a normal distribution. In the other case when using StandardScaler(), we can see on the probability plot of Fig. 26 that the data mostly fitted with the mean line, indicating that it was a normal distribution.

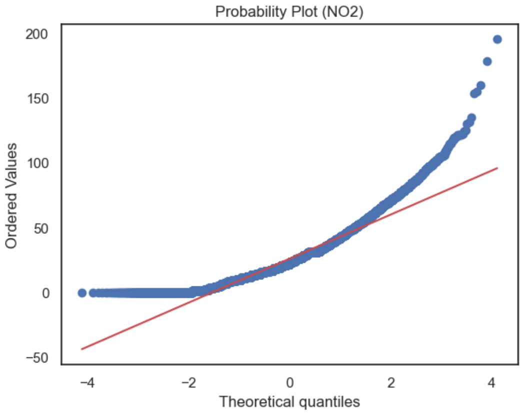


Fig. 25: Probability Plot for NO_2

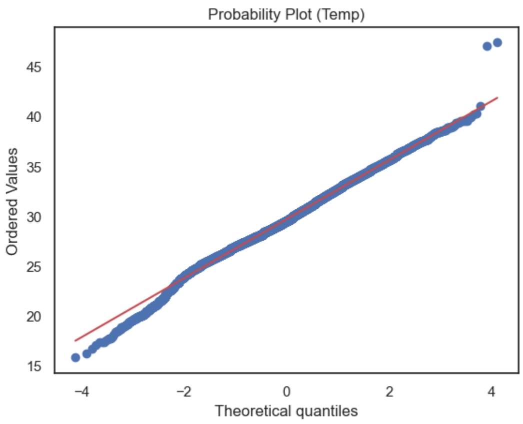


Fig. 26: Probability Plot for temperature

5) Encoding: In this data set, only the District is encoded. This does not have any order meaning e.g. Thonburi does not have any higher or lower hierarchy than Bang Khun Thian, so we'll encode with One-Hot Encoder.

6) Creating Preprocess Pipeline: This pipeline will include only Scaling and Encoding since replacing (filled) missing values was done in step before. We used ColumnTransformer() to ensure that each column receive the correct pre-process step.

TABLE I: Preprocess Pipeline Summary

Column Name	**Pipeline Step(s)**		
	Standard Scale	***MinMax Scale***	***OneHot Encoder***
CO	✓		
NO		✓	
NO2		✓	
NOX		✓	
O3		✓	
Windspeed		✓	
Winddir		✓	
Temp	✓		
Relhum	✓		
Rain	✓		
PM10[a]		✓	
District			✓

[a] PM.10 will be dropped in other scenario.

E. Modeling

The model we selected came from most in regression models available in scikit-learn.

1) Linear Regression will be used for basic prediction of PM2.5. It is the baseline for our goals. In addition, it is the most simple one among others. However, it can't be comprehended with outliers.
2) Ridge Regression
3) Lasso Regression
4) Gradient Boosting Regression
5) Histogram-based Gradient Boosting Regression Tree
6) ϵ-support Vector Regression (SVR)
7) K Neighbor Regression
8) Decision Tree Regression
9) Random Forest Regression
10) AdaBoost Regression

To select the best model for this purpose, we did k-fold cross-validation with k = 5. After we got the best model, we did a grid search with cross-validation to find the best hyper-parameters for that model with k-folds k = 10. Then, we gave the best model and best hyper-parameters to train on the whole training set again, including the validation set.

F. Training

Training will be done by the scikit-learn .fit method. We trained the best model with the best hyper-parameters with all training sets, unlike in cross-validation where some parts of the training set were separated into the validation sets.

G. Evaluation

After training the model, we plan to evaluate the model using Mean Squared Error (MSE) and R^2 score as shown in (3) and (4). R^2 score was selected because it can tell how much proportion of the whole data set this regression model explained. This evaluation process was nicely provided by scikit-learn for easy implementation directly from the pipeline we created. The overall process from selecting the model until the evaluation is shown in Fig. 27.

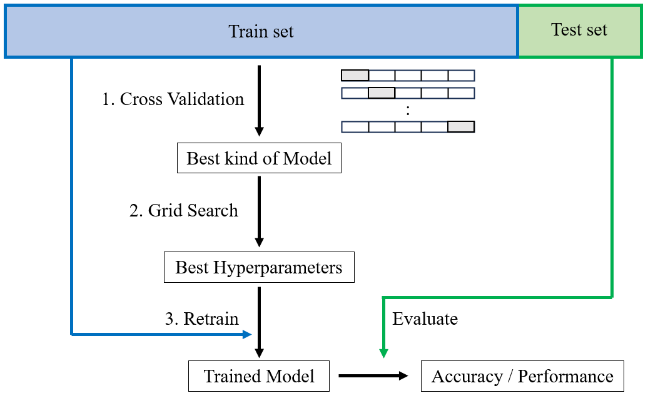


Fig. 27: Training Process

The evaluation matrix scores we tried to use are Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Squared Error (MSE), and Coefficient of Determination (R^2 score).

$$MAE = \frac{\sum_{i=1}^{m} |x_i - \hat{x_i}|}{m} \tag{1}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{m} (x_i - \hat{x_i})^2}{m}} \tag{2}$$

$$MSE = \frac{1}{m} \sum_{j=1}^{m} (x_i - \hat{x_i})^2 \tag{3}$$

$$R^2 = [\frac{1}{M} \frac{\sum_{j=1}^{M} [(Y_j - \overline{Y})(X_j - \overline{X})]}{\sigma_y \sigma_x}]^2 \tag{4}$$

where,

m and M are the number of observations
$\hat{x}_i$ is the predicted value
x_i is the actual value
σ_x is the standard deviation of the observation X
σ_y is the standard deviation of the observation Y
X_j is the observed values
$\overline{X}$ is the mean of the observed values
Y_j is the calculated values
$\overline{Y}$ is the mean of the calculated values.

H. Deployment

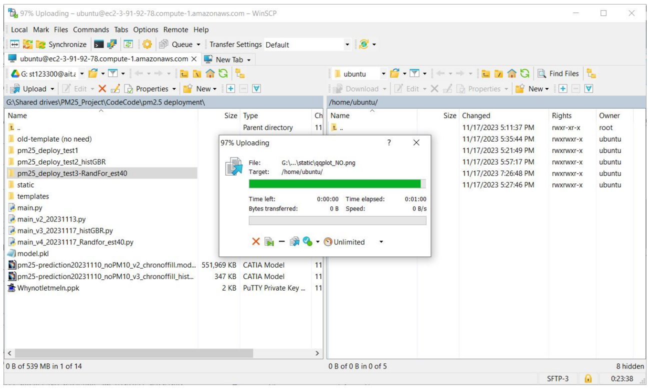
The deployment will use a microweb framework, Flask, and some basic HTML to show the results. The steps are as follows:

1) Deploy model in local host.
   a) Develop Machine Learning model.
   b) Convert model file to pickle object (.pkl) or joblib (.model).
   c) Create one Flask website app (app.py), tutorial from [13].
   d) In the Flask app, load the model object and create a form for user input.
   e) Create a prediction method in app.py to make a prediction.
   f) Test deployment on local host port 8080, tutorial from [15].
2) Upload all codes to Github https://github.com/BaiPorAndFern/CP_pm2.5_prediction.
3) Deploy the model on Amazon Web Service (AWS).
   a) Create an account.
   b) Create Administration-to-Customer (A2C) instance
   c) Edit security group in Inbound rules, Adding TCP port 8080.
   d) Download PuTTY private keygen (.ppk file).
   e) Download and install Putty [10] and WinSCP [11].
   f) Upload Flask website to EC2 using WinSCP. Note that before uploading, you must activate the key pair by double-clicking on it in Fig. 28.
   g) Install necessary packages on the virtual machine:
      - `sudo apt-get update`,
      - `sudo apt-get install python3-pip`, and
      - `pip3 install numpy flask scikit-learn`
   h) run the main.py `python main.py`.

The overall process for deployment is shown in Fig. 29.

VI. Results

This project uses a total of 11 features from 5 weather stations data set. After pre-processing was done, we began using some part of the data set, the train set, to select the model and its hyperparameters. In Fig. 30, the model selection result was shown. The result showed the validation R^2 score, ranked from highest to lowest. Hence, the Random Forest regression



97% Uploading – ubuntu@ec2-3-91-92-78.compute-1.amazonaws.com – WinSCP

Local Mark Files Commands Tabs Options Remote Help
Synchronize
Queue
Transfer Settings Default
ubuntu@ec2-3-91-92-78.compute-1.amazonaws.com
New Tab
G: st123300@ait.
Upload
Edit
Properties
New
ubuntu
Download
Find Files

G:\Shared drives\PM25_Project\CodeCode\pm2.5 deployment\

Name	Size	Type	Ch
..		Parent directory	11
old-template (no need)			
pm25_deploy_test1			
pm25_deploy_test2_histGBR			
pm25_deploy_test3-RandFor_est40			
static			
templates			
main.py			
main_v2_20231113.py			
main_v3_20231117_histGBR.py			
main_v4_20231117_Randfor_est40.py			
model.pkl			
pm25-prediction20231110_noPM10_v2_chronoffill.mod...	551,969 KB	CATIA Model	11
pm25-prediction20231110_noPM10_v3_chronoffill_hist...	347 KB	CATIA Model	11
Whynotletmeln.ppk	2 KB	PuTTY Private Key ...	11

0 B of 539 MB in 1 of 14

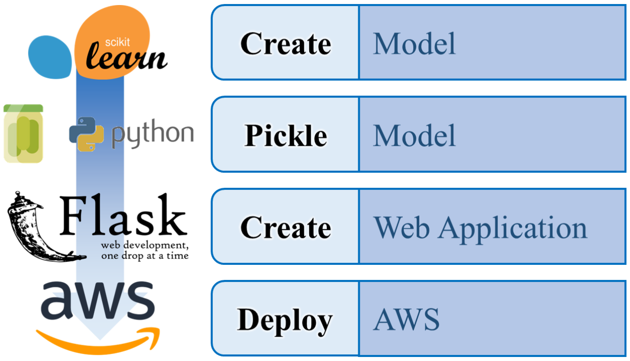
/home/ubuntu/

Name	Size	Changed	Rights	Owner
..		11/17/2023 5:11:37 PM	rwxr-xr-x	root
		11/17/2023 5:35:44 PM	rwxrwxr-x	ubuntu
		11/17/2023 5:21:49 PM	rwxrwxr-x	ubuntu
		11/17/2023 5:57:17 PM	rwxrwxr-x	ubuntu
		11/17/2023 7:26:48 PM	rwxrwxr-x	ubuntu
		11/17/2023 5:27:46 PM	rwxrwxr-x	ubuntu

0 B of 0 B in 0 of 5 8 hidden

97% Uploading
File: G:\...\static\qqplot_NO.png
Target: /home/ubuntu/
Time left: 0:00:00 Time elapsed: 0:01:00
Bytes transferred: 0 B Speed: 0 B/s

Fig. 28: Upload to AWS using WinSCP [11]



Create Model
Pickle Model
Create Web Application
Deploy AWS

Fig. 29: Deployment Steps

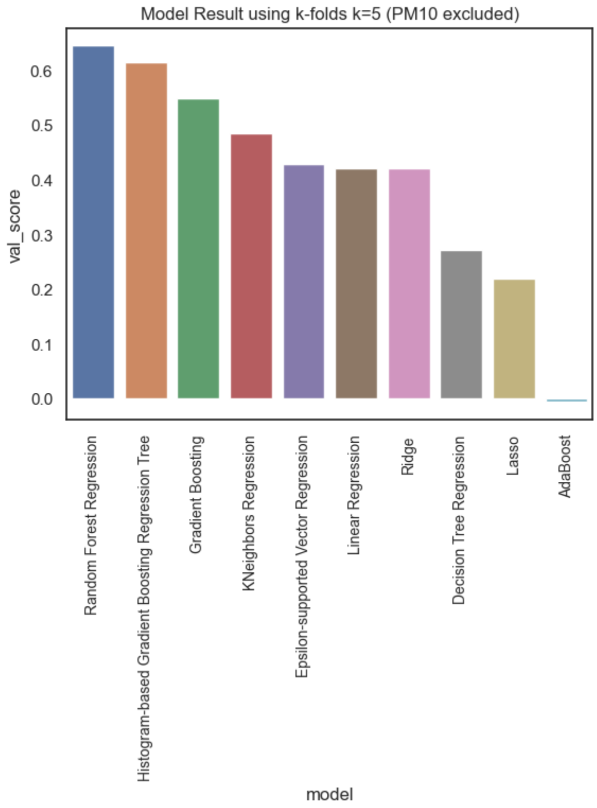


Fig. 30: Model Selection

was selected because of its performance with the highest R^2 score of 0.645.

We now have the algorithm to train. Then, we need to adjust its hyper-parameters to reach its optimal performance. The results from fine-tuning hyper-parameters in Fig. 31 showed that, among others, Random Forest Regression with `n_estimators` = 200 is the best with R^2 score of 0.648. However, it does not differ much from `n_estimators` = 40, but very different at the size of the model object. Thus, we select `n_estimators` = 40.

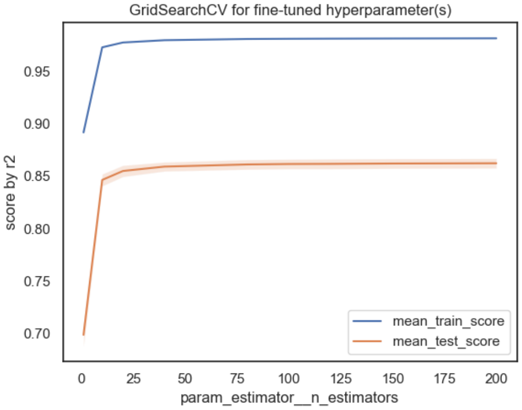


Fig. 31: Fine-tuned Hyper-parameter

After obtaining the optimal model possible, we trained it to the whole training set and evaluated using an unseen test set. The model shows the result of MSE and R^2 score of 162.97 and 0.443, respectively. In this step, we found out that the most important feature was NO_2. Results come from pipeline build-in attributes `feature_importances_` and all features rank is shown in Fig. 32.

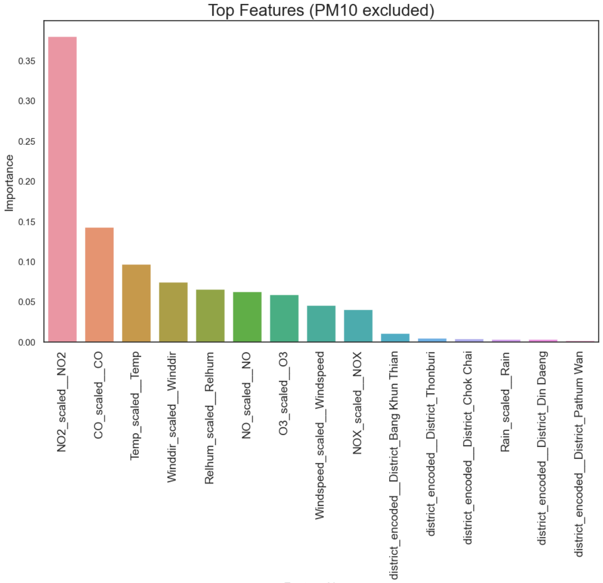
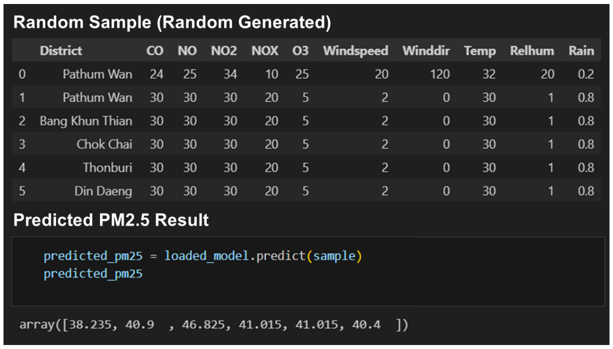


Fig. 32: Feature Importance

Then, after the model was trained, we tested on another sample of unseen data, excluded from our data set. The model showed that it can accept all input we gave to the model and made PM2.5 prediction results as shown in Fig. 33. We observed that, among other stations, Bang Khun Thian, if selected, will be sensitive to predicted PM2.5 the most. This can be traced back to feature importance in Fig. 32 that shows Bang Khun Thian is the most important district.



Random Sample (Random Generated)

	District	CO	NO	NO2	NOX	O3	Windspeed	Winddir	Temp	Relhum	Rain
0	Pathum Wan	24	25	34	10	25	20	120	32	20	0.2
1	Pathum Wan	30	30	30	20	5	2	0	30	1	0.8
2	Bang Khun Thian	30	30	30	20	5	2	0	30	1	0.8
3	Chok Chai	30	30	30	20	5	2	0	30	1	0.8
4	Thonburi	30	30	30	20	5	2	0	30	1	0.8
5	Din Daeng	30	30	30	20	5	2	0	30	1	0.8

Predicted PM2.5 Result

```
predicted_pm25 = loaded_model.predict(sample)
predicted_pm25
```

```
array([38.235, 40.9  , 46.825, 41.015, 41.015, 40.4  ])
```

Fig. 33: Test Model

VII. Conclusions and Future Works

In this project, we deploy the random forest regression model that can accept the new sample, and give a prediction of what pm2.5 will be, given the features. The model was initialized using a trained model from 5 weather stations in Bangkok.

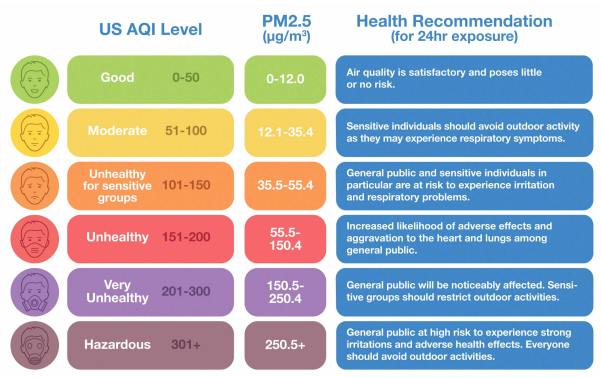
The main benefit of this project is predicting the PM2.5 value based on given weather conditions, specializing scikit-learn library to create a machine learning model to achieve that task, and deploying it to the website for end-users to use. This model can also be used with other station data too, to further enhance the performance of the model, making it able to predict various data from other places.

The future works can be that the model used does not include time series. This can be improved by including a Timestamp in the analytics and using a time-series algorithm like an Auto-Regressive Integrated Moving Average (ARIMA). Since the problem involves forecast with time series, this model can be used.

Also, We can add the Air Quality Index (AQI) based on the Thailand Air Pollution Department criterion as shown in Fig. 34 into prediction results and add recommendations related to the predicted PM2.5 value, giving the users more on how should they act and prepared (Decision-Making).

Another thing that can be improved is changing the evaluation score. Using R^2 score alone can sometimes mislead the users if more features are added. R^2 score will always increase if more features are added, regardless of whether these features are significant or not. So using the adjusted R^2 score would be more adjusted to the real scenario. Now, adding an unnecessary variable will often decrease the adjusted R^2 score as there are now indicators of non-significant terms included in the model.

Lastly, including more weather data and PM2.5 from other stations and other provinces would give more in-depth details on PM2.5 data in Thailand and will strengthen the model further in predicting PM2.5 in more diverged places.



US AQI Level		PM2.5 (µg/m^3)	Health Recommendation (for 24hr exposure)
Good	0-50	0-12.0	Air quality is satisfactory and poses little or no risk.
Moderate	51-100	12.1-35.4	Sensitive individuals should avoid outdoor activity as they may experience respiratory symptoms.
Unhealthy for sensitive groups	101-150	35.5-55.4	General public and sensitive individuals in particular are at risk to experience irritation and respiratory problems.
Unhealthy	151-200	55.5-150.4	Increased likelihood of adverse effects and aggravation to the heart and lungs among general public.
Very Unhealthy	201-300	150.5-250.4	General public will be noticeably affected. Sensitive groups should restrict outdoor activities.
Hazardous	301+	250.5+	General public at high risk to experience strong irritations and adverse health effects. Everyone should avoid outdoor activities.

Fig. 34: US AQI levels, equivalent PM2.5 standards by $\mu g/m^3$, and health recommendations for each level. [3]

References

[1] Leah Luyen, “EDA, Data Preprocessing, Feature Engineering: We are different!”, Medium.com, (access October 23, 2023).

[2] Pollution Control Department of Thailand, “Particulate Matter Historical Data”, http://air4thai.pcd.go.th/webV2/history/ (access October 16, 2023).

[3] https://www.iqair.com/th-en/newsroom/thailand-2021-burning-season

[4] Mehdipour V., Stevenson D.S., Memarianfard M. et al, “Comparing different methods for statistical modeling of particulate matter in Tehran, Iran”, Air Quality Atmosphere Health 11, pp.1155—1165 (2018), https://doi.org/10.1007/s11869-018-0615-z

[5] https://scikit-learn.org/stable/

[6] Nidhi Sharma, Shweta Taneja, Vaishali Sagar, and Arshita Bhatt, “Forecasting air pollution load in Delhi using data analysis tools”, Procedia Computer Science, vol. 132. 2018. pp.1077–1085. https://www.sciencedirect.com/science/article/pii/S1877050918307555.

[7] Doreswamy, Harishkumar K. S., Yogesh K.M. and Ibrahim Gad. “Forecasting Air Pollution Particulate Matter (PM2.5) Using Machine Learning Regression Models”. Procedia Computer Science. vol. 171. 2020. pp. 2057–2066. https://www.sciencedirect.com/science/article/pii/S1877050920312060

[8] https://www.freepik.com/vectors/pm2-5

[9] https://www.bangkokhospital.com/en/content/5-ways-to-deal-with-toxic-dust-should-be-shared

[10] https://www.putty.org/

[11] https://winscp.net/eng/download.php

[12] “How to calculate the adjusted r2 value using scikit” , https://stackoverflow.com/questions/51038820/how-to-calculated-the-adjusted-r2-value-using-scikit (access November 15, 2023).

[13] “Learn Flask for Python - Full Tutorial”, https://www.youtube.com/watch?v=Z1RJmh_OqeA&t=518s (access November 9, 2023).

[14] “Rails Server in EC2”, https://www.youtube.com/watch?v=bTSopMRFfhI (access November 1, 2023).

[15] “Deploy Machine Learning Model Flask”, https://www.youtube.com/watch?v=MxJnR1DMmsY (access November 9, 2023).

[16] “How to Deploy a Flask App to Heroku”, https://www.youtube.com/watch?v=D2GLVoiEZyE (access November 10, 2023).