Forecasting Disease Spread

Dengue fever is a mosquito-borne disease that occurs in tropical and sub-tropical parts of the world. In mild cases, symptoms are similar to the flu: fever, rash, and muscle and joint pain. In severe cases, dengue fever can cause severe bleeding, low blood pressure, and even death. Using environmental data collected by various U.S. Federal Government agencies—from the Centers for Disease Control and Prevention to the National Oceanic and Atmospheric Administration in the U.S. Department of Commerce, this project is trying to predict the number of dengue fever cases reported each week in San Juan, Puerto Rico and Iquitos, Peru.

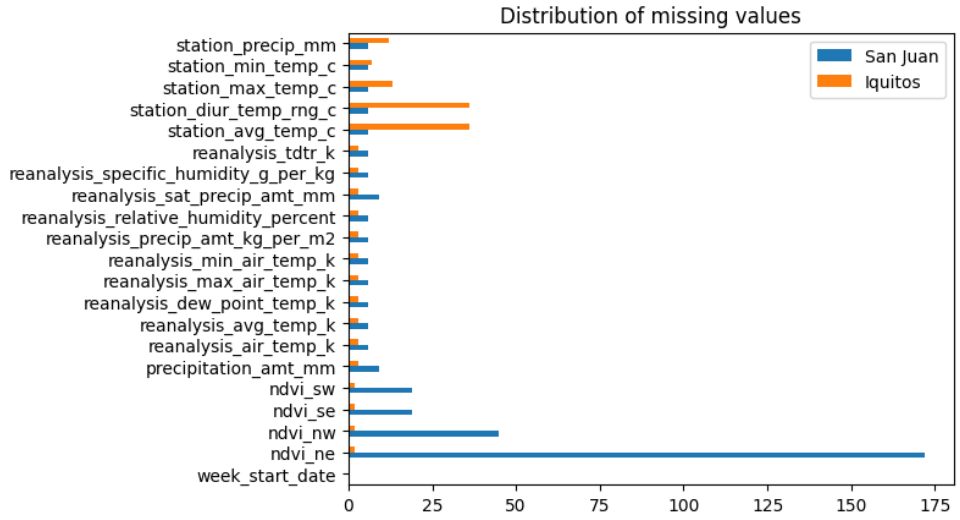
Explore Data Analysis

The training dataset is a relatively small dataset containing only around 1400 rows. We have initially 24 features as follows:

Column Name	Type
city	String
year	Integer
week_start_date	Timestamp
station_max_temp_c	Float
station_avg_temp_c	Float
station_precip_mm	Float
station_min_temp_c	Float
station_diur_temp_rng_c	Float
precipitation_amt_mm	Float
reanalysis_sat_precip_amt_mm	Float
reanalysis_dew_point_temp_k	Float
reanalysis_air_temp_k	Float
reanalysis_relative_humidity_percent	Float
reanalysis_specific_humidity_g_per_kg	Float
reanalysis_precip_amt_kg_per_m2	Float
reanalysis_max_air_temp_k	Float
reanalysis_min_air_temp_k	Float
reanalysis_avg_temp_k	Float
reanalysis_tdtr_k	Float
ndvi_ne	Float
ndvi_nw	Float
ndvi_se	Float
ndvi_sw	Float

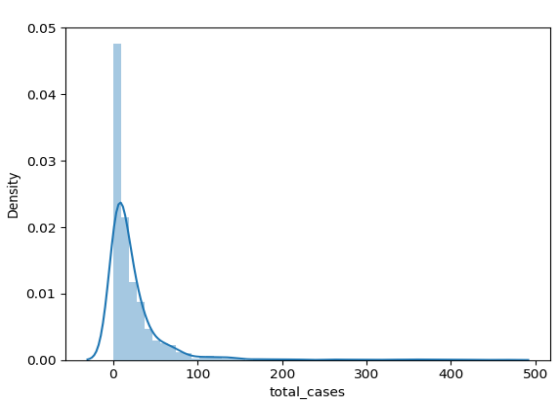
More details about the meaning of features can be found at
https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/page/82/#features_list

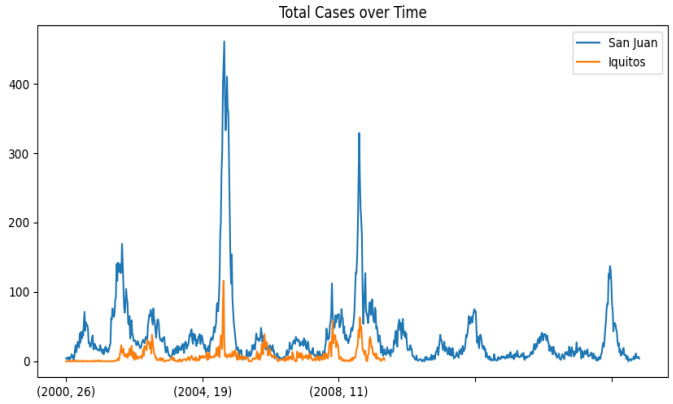
The first thing is to check how many missing values we have. The result is as follows:



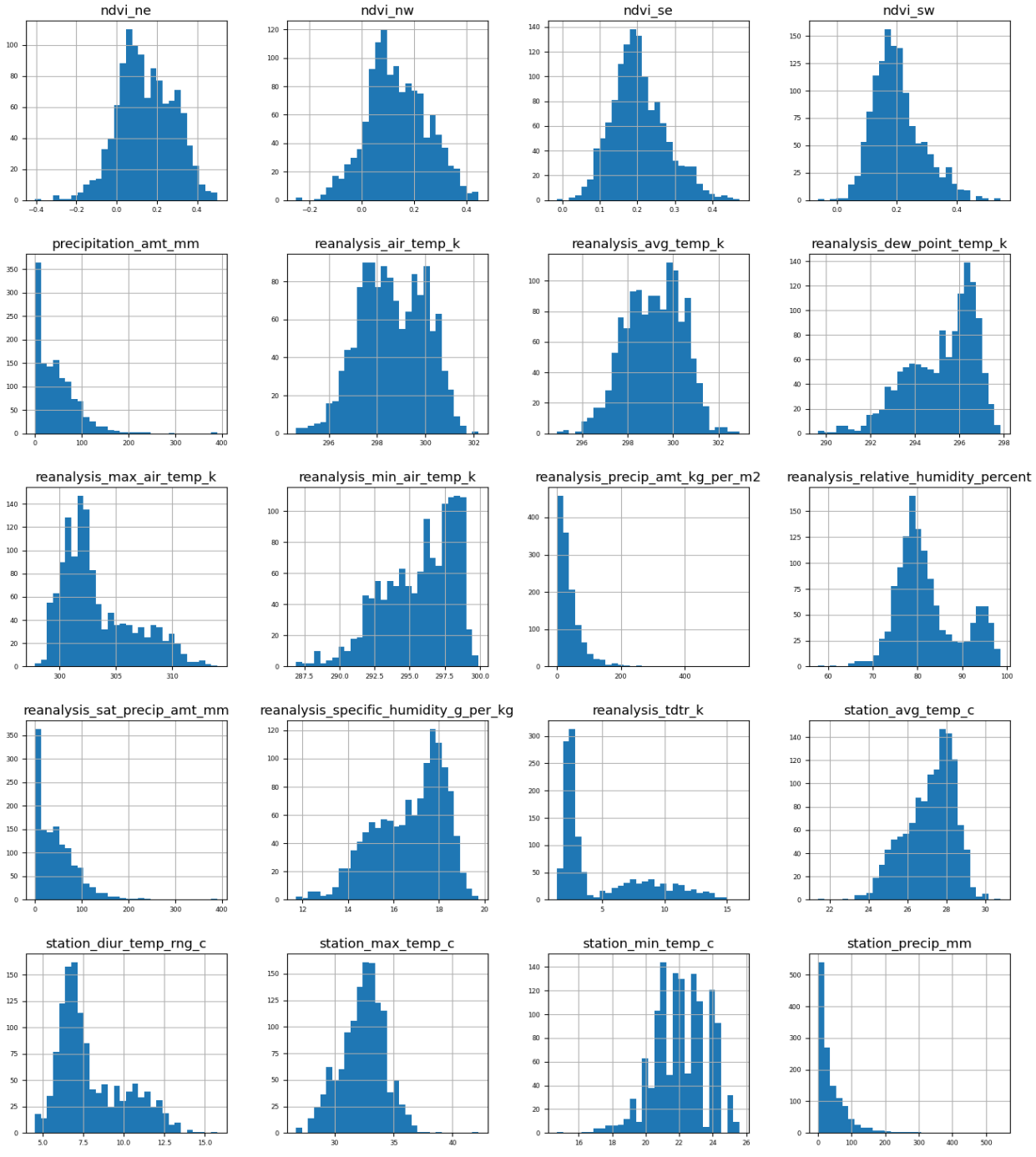
From the plot we can see that most features have very few missing values. Ndvi_ne column has around 13% missing values which mainly sources from San Juan. We can remove this feature for further step of feature engineering.

The distribution of target value is right skewed as the following graph shows. This means that we can do a log transformation for the target value. If we plot the total cases with time goes, we can see that San Juan has a longer timeline compared with Iquitos (2:1 for the size of data).





First, we checked the distribution of features for the data.



As we observed, a few features are skewed, and we can use different methods of transformation to transform the features. The decision is as follows:

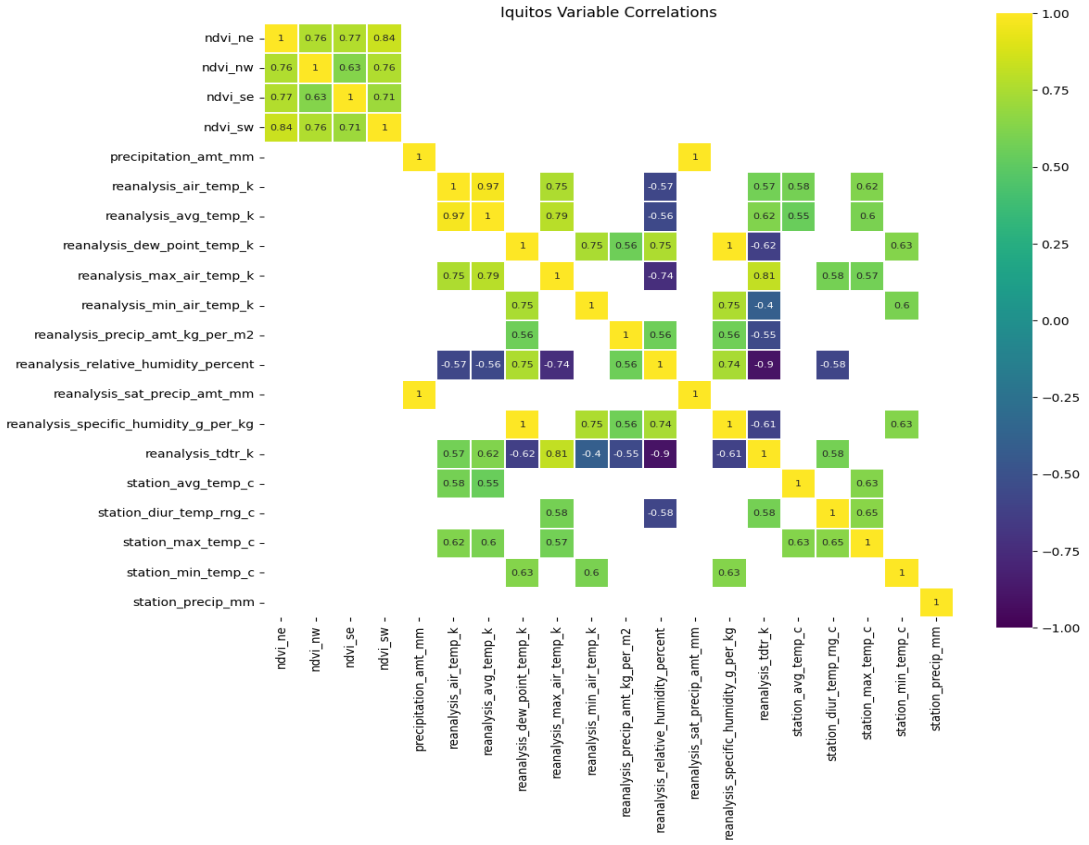
Right-skewed features	**Left-skewed features**
precipitation_amt_mm	reanalysis_min_air_temp_k
reanalysis_precip_amt_kg_per_m2	reanalysis_specific_humidity_g_per_kg
reanalysis_sat_precip_amt_mm	station_min_temp_c
station_precip_mm	station_avg_temp_c
reanalysis_tdtr_k	

For right-skewed features, we use log transformation while for left-skewed features, we use box-cox transformation. One interesting finding is that precipitation features are always right skewed.

Some insights can also be found from correlation plots for two cities respectively.

San Juan:

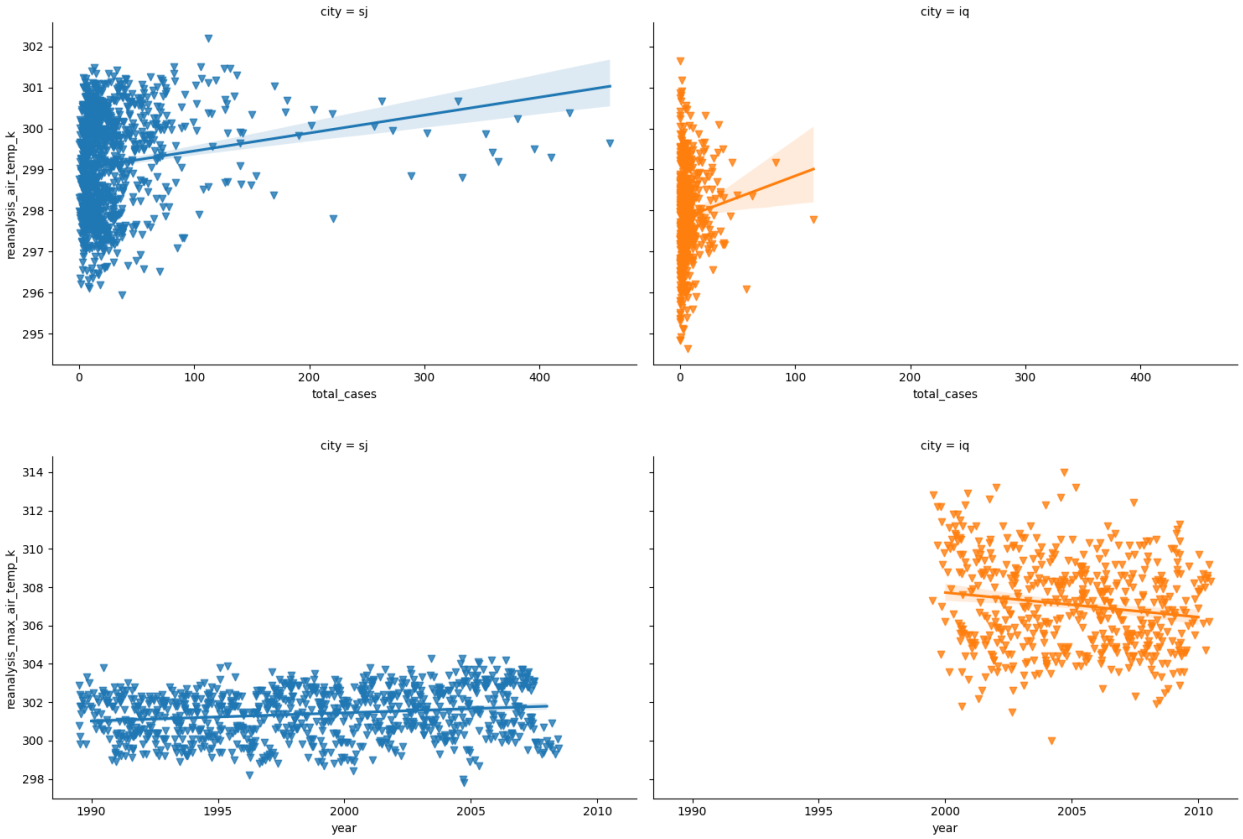
Iquitos:




```
There is 5 strongly correlated values with number of total cases:
reanalysis_min_air_temp_k    0.325252
station_min_temp_c           0.267109
reanalysis_air_temp_k        0.264952
weekofyear                   0.216452
reanalysis_avg_temp_k        0.151637
```

Features have relatively weak correlation with total cases (strongest is just 32.5%). However, climate variables have much stronger correlation with each other such as reanalysis_dew_point_temp_k and reanalysis_specific_humidity_g_per_kg. More details would be discussed in the part of feature engineer.

We also have many plots for distribution between features and target values. And all plots are in the EDA notebook, for conciseness, I will not show all but give some observations here.



- For each category of features, we can see that two cities have relatively different patterns for the distribution between features and target values. This means that we potentially can model two cities respectively.
- We can see few features have strong trend with time going so that the column of year is not necessary in the modelling.

Feature Engineer & Selection

Feature engineering and selection is always one of the most important parts before we put some real models for the data. EDA has already given us enough insights into the data and now we need to use some techniques to extract valuable information from the features.

Feature transformation:

Log Transformation	Square Transformation
precipitation_amt_mm	reanalysis_dew_point_temp_k
reanalysis_precip_amt_kg_per_m2	reanalysis_min_air_temp_k
reanalysis_sat_precip_amt_mm	reanalysis_relative_humidity_percent
station_precip_mm	station_min_temp_c

Also, the city columns should be encoded to 1 as San Juan and 0 as Iquitos.

Feature Removal:

Features	Reason
reanalysis_avg_temp_k	reanalysis_avg_temp_k and reanalysis_air_temp_k is literally similar in meaning and indeed they are highly correlated.
reanalysis_sat_precip_amt_mm	reanalysis_sat_precip_amt_mm is almost the same (values) as precipitation_amt_mm.
ndvi_ne	13% missing
year	Few features have strong trend as time goes
weekofyear	Not useful compared with month
week_start_date	Month is more useful for the exact date

Feature Generation:

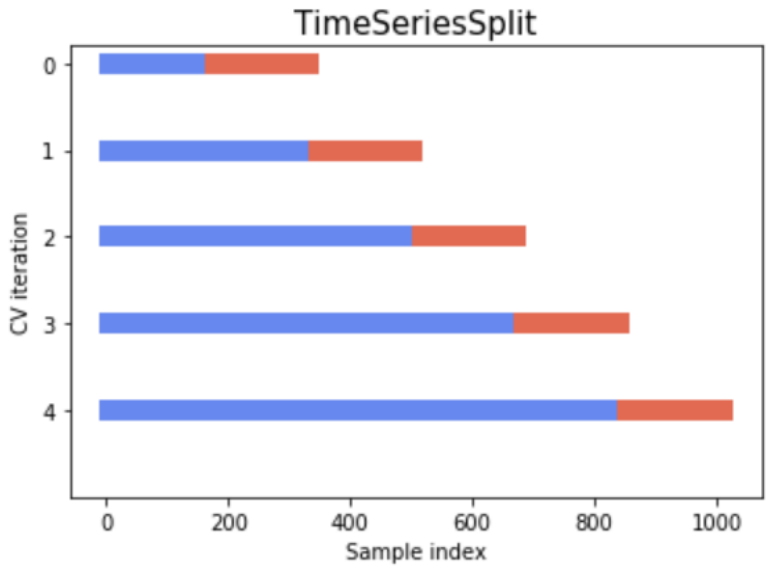
Features	Methodology	Reason
month	Extract month from week_start_date	Strong indicators
station_precip_mm_rolling_month_mean	Rolling mean (4 weeks, around one month) of station_precip_mm	Precipitation might be more useful for a longer-period window
station_precip_mm_diff	Difference between this week precipitation and last week's	Difference of precipitation reflect the change of climate
reanalysis_temp_humid_index	Interaction between reanalysis_relative_humidity_percent and reanalysis_tdtr_k	Hot and wet weather trends to increasing total cases of dengue fever
ndvi_max_precip	Interaction between max(ndvi features) and precipitation_amt_mm	More vegetation combining with more precipitation may lead to increasing total cases of dengue fever

Finally, I used linear interpolation to fill missing values.

Single Model

Since the size of dataset is relatively small, data is divided into 90% of training data and 10% for validation data and I prefer to focus on tree-based models. The baseline models is the random model by just using mean value of train labels to predict validation labels, and the MAE is 20.37. The other two common baseline models would be linear regression and random forest. Without any hyperparameter tuning, linear regression has a MAE of 17.5 for validation dataset and random forest has a MAE of 13.03 for validation dataset.

For more complicated tree-based models, I chose LightGBM and XGBoost. I applied time series split cross validation to reduce the variance of model performance due to the limited size of dataset. Instead of traditional K-fold cross validation, time series split cross validation also avoids the issue of data leakage. The objective score to minimize is the sum of MAE mean and standard deviation to reduce both bias and variance of the model.

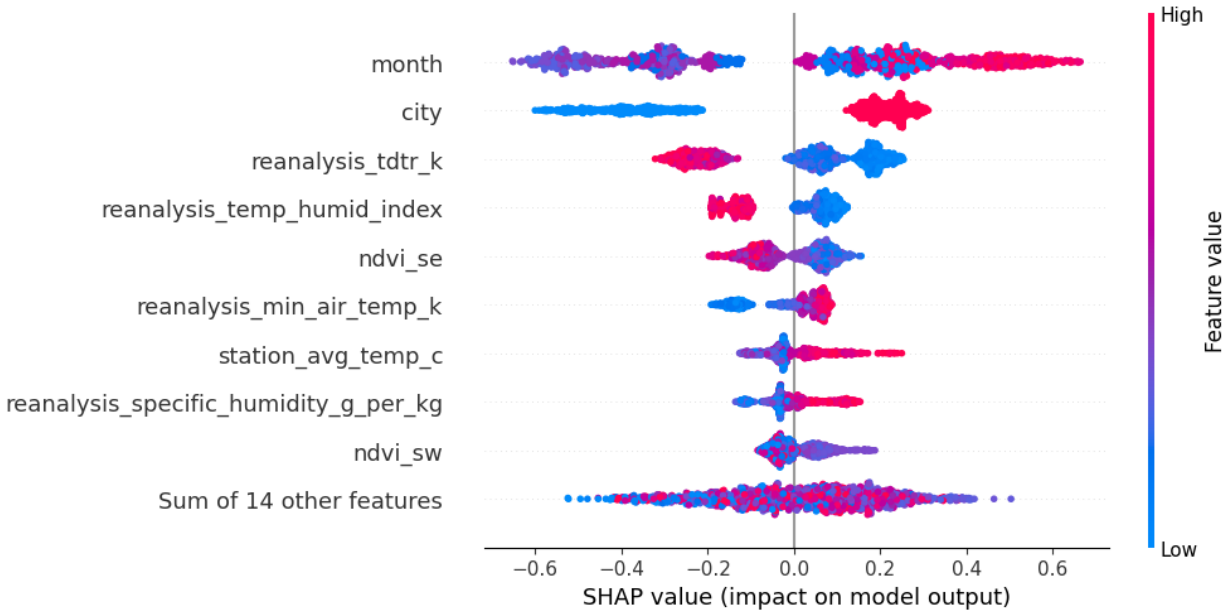


For the hyperparameter tuning, I relied on Optuna framework, which is state-of-the-art algorithms to choose the best parameters and easy to visualize the process of tuning. For LightGBM, I mainly focused on tuning learning rate, max depth of the trees and fraction of features. A similar process applied to XGBoost but with different names of parameters. The result is that both XGBoost and LightGBM outperform the baseline model and LightGBM has the best MAE of 12.05 (XGBoost has the MAE of 12.54).

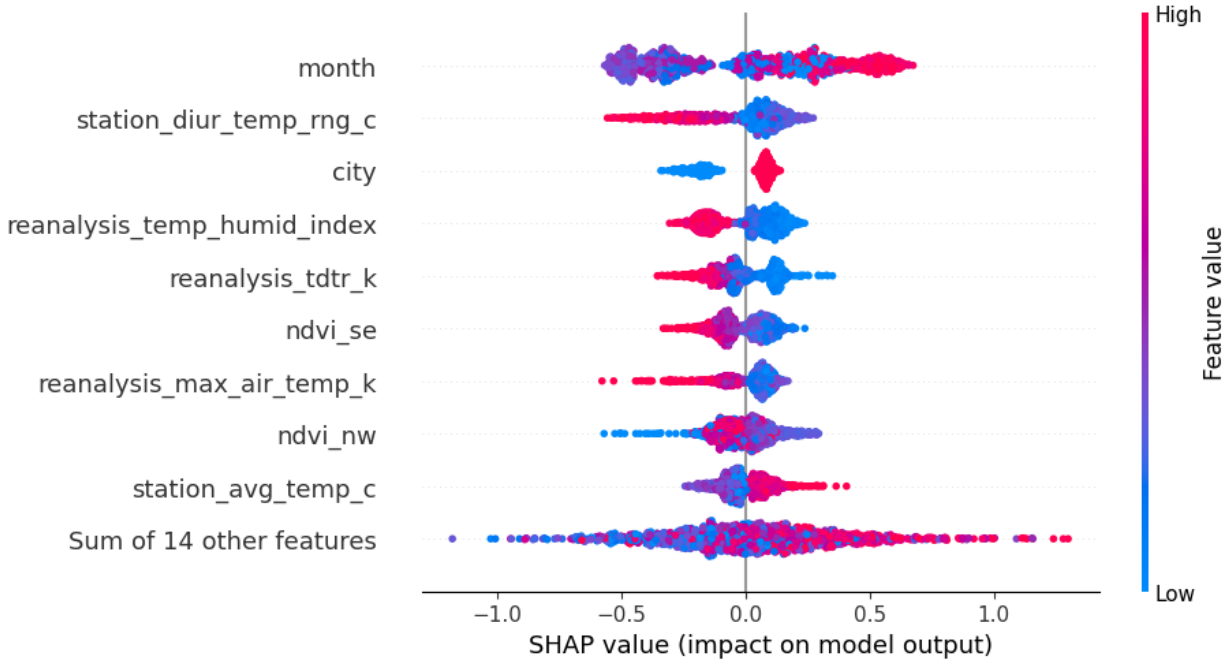
Feature Importance Analysis

For the final part of the project, we should focus on how the model values all the features. I relied on shap to analyze the feature importance with some nice visualizations.

LightGBM



XGBoost



It looks like the two algorithms share the same pool of most important features while they rank these features differently. Our new-generated feature temp_humidity_index (interaction between temperature and humidity) and month are included as one of the most important features, which means that the month and the hotter & wetter weather can reveal some information of the occurrence of disease.

Models by Locations

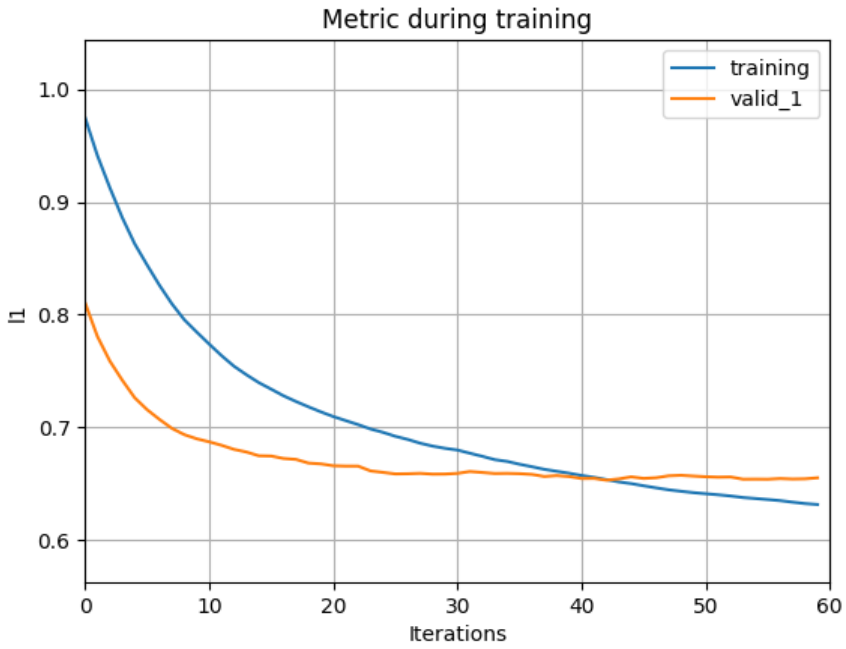
I did some experiments on training separate models for San Juan and Iquitos but there are no obvious improvements happening (a little decrease indeed). The potential reasons are that single model also treats the information of city as very important and for each city, the tiny size of data also limits the improvement space of separate model. I also tried negative binomial regression due to the right skew distribution of target values for training data for two cities respectively while the result was not ideal as well (MAE of 16.88).

Next Steps

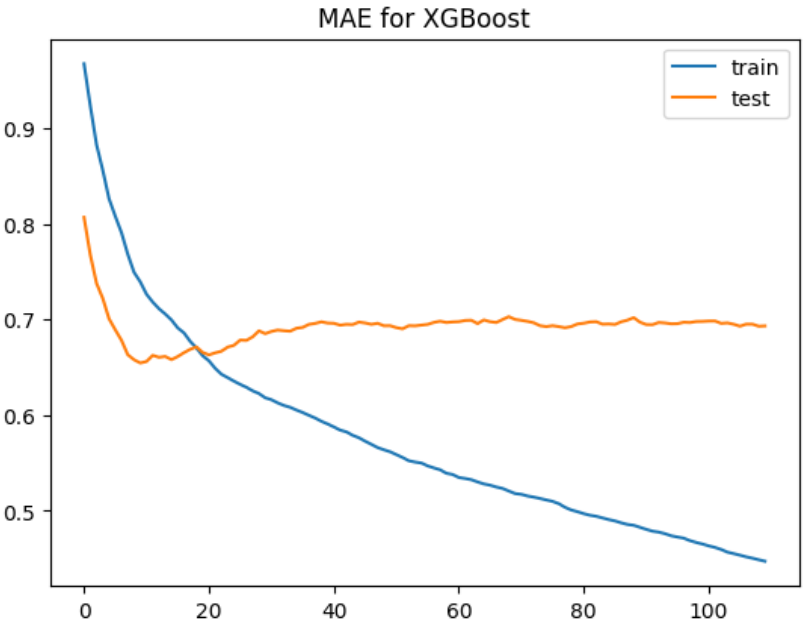
- There is space for more sophisticated feature engineering and selection. We can use tsfresh (https://github.com/blue-yonder/tsfresh) to make various time-related features and featurewiz to automatically generate new features and select most strong features from the pool. However, when we rely on these packages, we must make sure we understand the full meaning of new-generated features.
- We can make the ensemble of different models to increase the final performance. This method is a little Kaggle-style, and we will have to decide the weight of each model (we can train a separate model to decide the weights).
- Since there is no time for me to go further about the model by locations and potentially, we can search for more possibilities for this direction in future as well.

Appendix:

MAE plot for training and validation for LightGBM



MAE plot for training and validation for XGBoost



For lightGBM, it looks like we have reasonable group of hyperparameters and no obvious indication of overfitting. For XGBoost, although training loss is still decreasing, the validation loss is not improving anymore so any more epochs of training will lead to overfitting.