



DengAI: Predict Disease Spread

Dengue is a mosquito-borne disease that occurs in tropical and sub-tropical parts of the world. In this study, let's predict the total cases in San Juan, Puerto Rico & Iquitos, Peru to prevent the disease spread.

Shuting Zhang



Approach to solve this problem



Predict the
number of
dengue fever

1 Project Background, what's the question?

2 Exploratory data analysis (EDA)

3 Data Preprocessing

4 Modeling

5 Conclusion and what can be improved?

6 References



Step 1: understand the question

1 Project Background, what's the question?

Predict the
number of
dengue fever



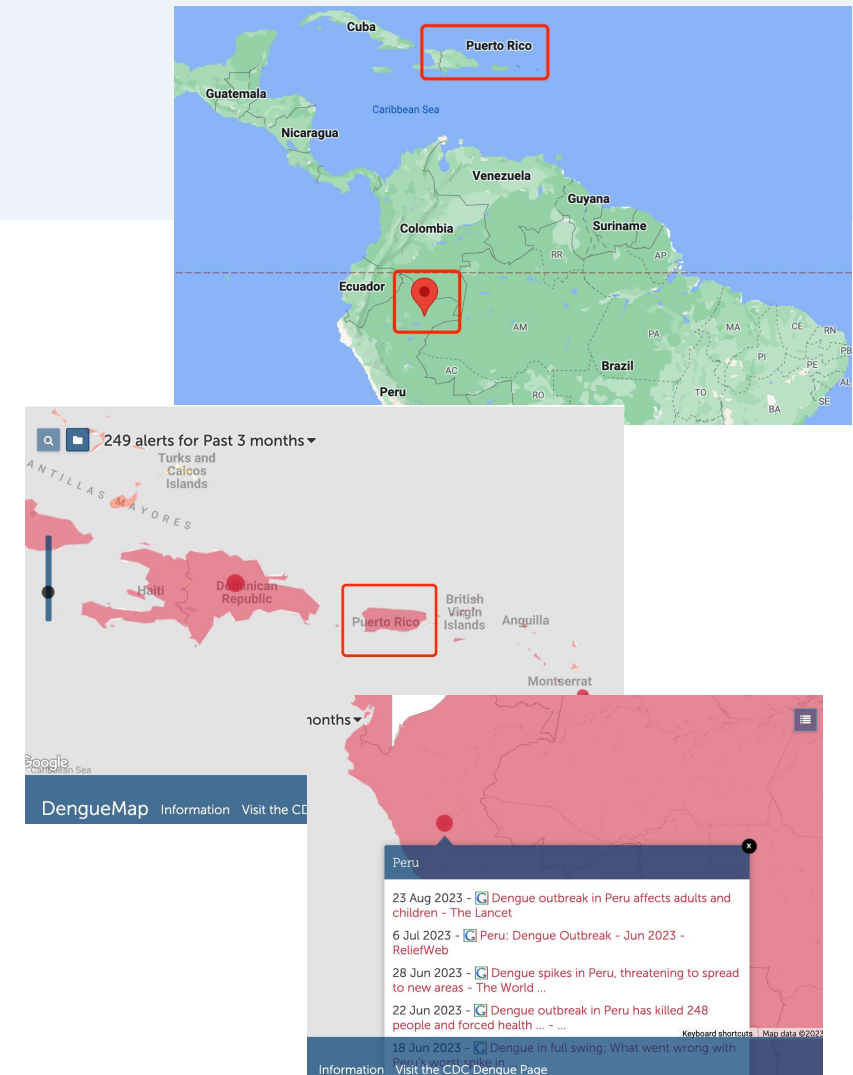
Goal: Predict the Next Pandemic Initiative



- **Dengue 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. But severe cases are dangerous, and **dengue fever can cause severe bleeding, low blood pressure and even death.**
- Dengue is carried by mosquitoes, the transmission dynamics of dengue are related to climate variables such as temperature and precipitation.

In this challenge:

- I used provided climate variables to predict dengue cases in **San Juan, Puerto Rico & Iquitos, Peru**



Step 2 - EDA

Predict the
number of
dengue fever

1 Project Background, what's the question?

2 Exploratory data analysis (EDA)

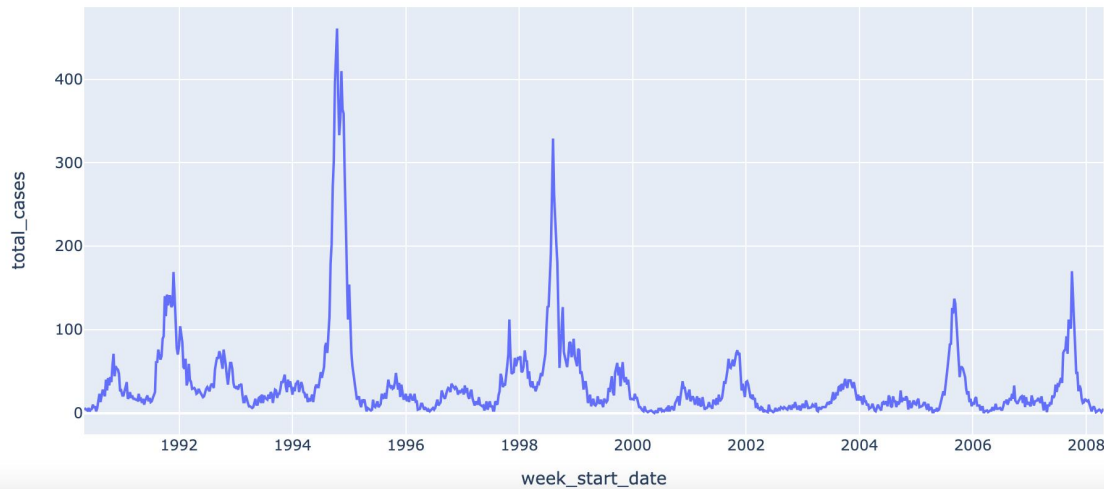




Dengue total cases in San Juan & Iquitos

Goal: predict the `total_cases` label for each (city, year, weekofyear) in the test set for these 2 cities, test data for each city are 5 and 3 years respectively

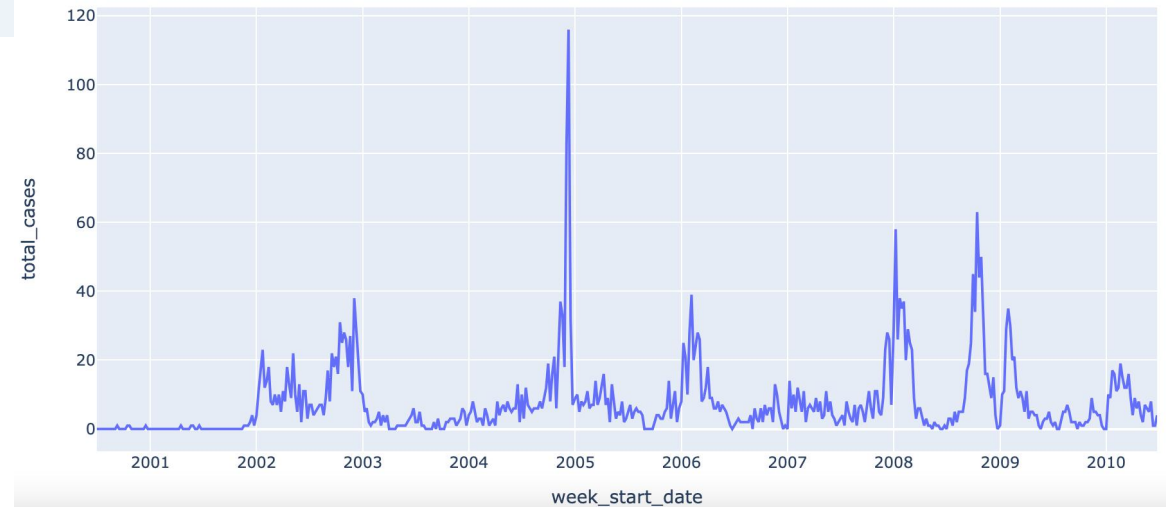
SJ historical cases



Dengue cases in San Juan

- Historical data is from 1990 - 2008
- Test data: 2008 - 2013
- Cases in San Juan are more than Iquitos, looks more have a seasonality trend

IQ historical cases



Dengue cases in Iquitos

- Historical data is from 2000 - 2010
- Test data: 2010 - 2013
- The Dengue breakouts seem more random than San Juan, no visible seasonality or cycling



Features - Definitions

- **Precipitation values**

- station_precip_mm – Total precipitation
- precipitation_amt_mm – Total precipitation
- reanalysis_sat_precip_amt_mm – Total precipitation
- reanalysis_precip_amt_kg_per_m2 – Total precipitation

- **Temperature**

- station_max_temp_c – Maximum temperature
- station_min_temp_c – Minimum temperature
- station_avg_temp_c – Average temperature
- station_diur_temp_rng_c – Diurnal temperature range
- reanalysis_air_temp_k – Mean air temperature
- reanalysis_dew_point_temp_k – Mean dew point temperature
- reanalysis_max_air_temp_k – Maximum air temperature
- reanalysis_min_air_temp_k – Minimum air temperature
- reanalysis_avg_temp_k – Average air temperature
- reanalysis_tdtr_k – Diurnal temperature range

- **Humidity**

- reanalysis_relative_humidity_percent – Mean relative humidity
- reanalysis_specific_humidity_g_per_kg – Mean specific humidity

- **Satellite vegetation - Normalized difference vegetation index (NDVI)**

- ndvi_se – Pixel southeast of city centroid
- ndvi_sw – Pixel southwest of city centroid
- ndvi_ne – Pixel northeast of city centroid
- ndvi_nw – Pixel northwest of city centroid

Total 20 featurers

As we can see from the names, they are very similar.

For instance, “Total precipitation” has 4 variables, more explorations on similar variables are needed later to figure out if all of them should be use in our prediction



Features - Value Exploration

	count	mean	std	min	25%	50%	75%	max
year	1456.0	2001.031593	5.408314	1990.000000	1997.000000	2002.000000	2005.000000	2010.000000
weekofyear	1456.0	26.503434	15.019437	1.000000	13.750000	26.500000	39.250000	53.000000
ndvi_ne	1262.0	0.142294	0.140531	-0.406250	0.044950	0.128817	0.248483	0.508357
ndvi_nw	1404.0	0.130553	0.119999	-0.456100	0.049217	0.121429	0.216600	0.454429
ndvi_se	1434.0	0.203783	0.073860	-0.015533	0.155087	0.196050	0.248846	0.538314
ndvi_sw	1434.0	0.202305	0.083903	-0.063457	0.144209	0.189450	0.246982	0.546017
precipitation_amt_mm	1443.0	45.760388	43.715537	0.000000	9.800000	38.340000	70.235000	390.600000
reanalysis_air_temp_k	1446.0	298.701852	1.362420	294.635714	297.658929	298.646429	299.833571	302.200000
reanalysis_avg_temp_k	1446.0	299.225578	1.261715	294.892857	298.257143	299.289286	300.207143	302.928571
reanalysis_dew_point_temp_k	1446.0	295.246356	1.527810	289.642857	294.118929	295.640714	296.460000	298.450000
reanalysis_max_air_temp_k	1446.0	303.427109	3.234601	297.800000	301.000000	302.400000	305.500000	314.000000
reanalysis_min_air_temp_k	1446.0	295.719156	2.565364	286.900000	293.900000	296.200000	297.900000	299.900000
reanalysis_precip_amt_kg_per_m2	1446.0	40.151819	43.434399	0.000000	13.055000	27.245000	52.200000	570.500000
reanalysis_relative_humidity_percent	1446.0	82.161959	7.153897	57.787143	77.177143	80.301429	86.357857	98.610000
reanalysis_sat_precip_amt_mm	1443.0	45.760388	43.715537	0.000000	9.800000	38.340000	70.235000	390.600000
reanalysis_specific_humidity_g_per_kg	1446.0	16.746427	1.542494	11.715714	15.557143	17.087143	17.978214	20.461429
reanalysis_tctr_k	1446.0	4.903754	3.546445	1.357143	2.328571	2.857143	7.625000	16.028571
station_avg_temp_c	1413.0	27.185783	1.292347	21.400000	26.300000	27.414286	28.157143	30.800000
station_diur_temp_rng_c	1413.0	8.059328	2.128568	4.528571	6.514286	7.300000	9.566667	15.800000
station_max_temp_c	1436.0	32.452437	1.959318	26.700000	31.100000	32.800000	33.900000	42.200000
station_min_temp_c	1442.0	22.102150	1.574066	14.700000	21.100000	22.200000	23.300000	25.600000
station_precip_mm	1434.0	39.326360	47.455314	0.000000	8.700000	23.850000	53.900000	543.300000

Data Exploration

1. There are missing values
2. Only NVDI are normalized
3. Data in different scales, in later modeling, consider normalization or scaling
4. Week number 53 is a mistake, there are only 52 weeks in a year, need fix the wrong values
5. Temperature variables are in different units (K or C)



Feature visualization - samples

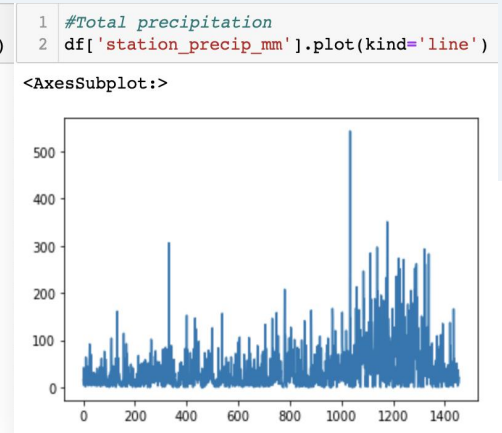
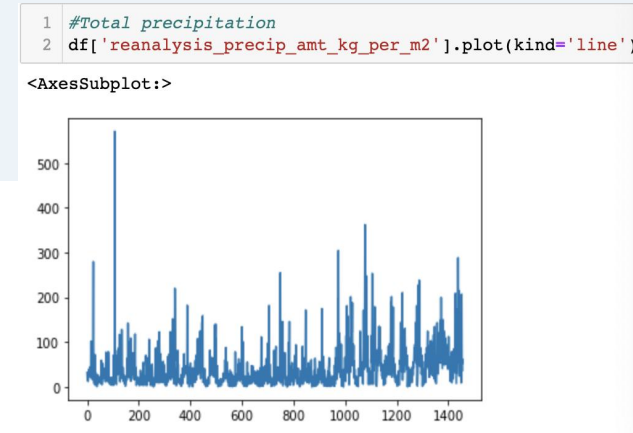
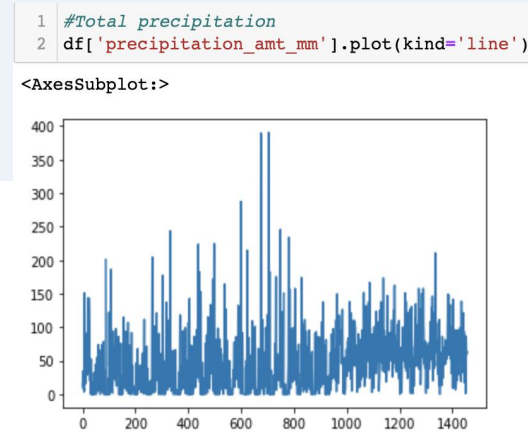
Observations

- The available data are all climate related data collected from different sources.
- Two different cities have very different trends
- Some data are collected from different organization, they seem to be highly correlated (r.g. Diurnal temperature range)
- Some data are quite different, even they are with same definitions (e.g. Total precipitation)

Next step: Data Processing

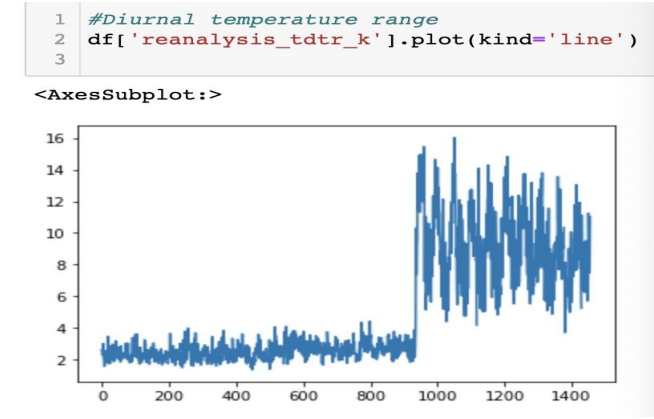
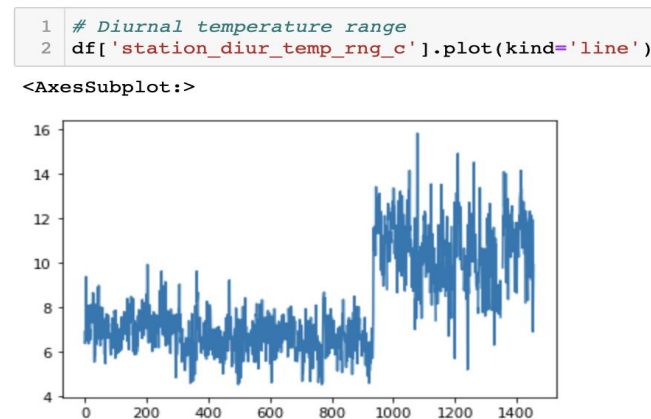
- check the data quality, e.g. outliers
- if there are NAs, fill NAs
- check if scaling/normalization is needed

Total precipitation



Diurnal temperature range

variation between a high air temperature and a low temperature that occurs during the same day



Step 3: Data Processing before modeling

Predict the
number of
dengue fever

1 Project Background, what's the question?

2 Exploratory data analysis (EDA)

3 Data Processing before modeling



Data Processing - Missing values

	column_name	percent_missing
city	city	0.00
year	year	0.00
weekofyear	weekofyear	0.00
week_start_date	week_start_date	0.00
ndvi_ne	ndvi_ne	13.32
ndvi_nw	ndvi_nw	3.57
ndvi_se	ndvi_se	1.51
ndvi_sw	ndvi_sw	1.51
precipitation_amt_mm	precipitation_amt_mm	0.89
reanalysis_air_temp_k	reanalysis_air_temp_k	0.69
reanalysis_avg_temp_k	reanalysis_avg_temp_k	0.69
reanalysis_dew_point_temp_k	reanalysis_dew_point_temp_k	0.69
reanalysis_max_air_temp_k	reanalysis_max_air_temp_k	0.69
reanalysis_min_air_temp_k	reanalysis_min_air_temp_k	0.69
reanalysis_precip_amt_kg_per_m2	reanalysis_precip_amt_kg_per_m2	0.69
reanalysis_relative_humidity_percent	reanalysis_relative_humidity_percent	0.69
reanalysis_sat_precip_amt_mm	reanalysis_sat_precip_amt_mm	0.89
reanalysis_specific_humidity_g_per_kg	reanalysis_specific_humidity_g_per_kg	0.69

Most of the features are having missing values. For instance, ndvi_ne is missing 13.3% of all values.

Method to fillnas in this project:

1. forwardfill/backwardfill – ffill/bffill :

- ffill is more appropriate when the missing values are at the beginning of the dataset, while bfill is more appropriate when the missing values are at the end of the dataset

- There are a lot of climate variables values: if fill by Mean/Median, it might not make sense, since temperature varies with datatype data, the values should be similar with the most recent value around to that point

2. use Interpolate() Method with limit parameter e.g linear



Data Processing - Standardizing & Normalization

Some data science knowledge:

- Variables that are measured at different scales do not contribute equally to the model which might lead to a poor performance in models.
- Normally, all the tree-based algorithms don't need data scaling or normalization
 - e.g. CART, Random Forests, Gradient Boosted Decision Trees .
- Normalization is a good technique to use when you do not know the distribution of your data or when you know the distribution is not Gaussian (a bell curve).
 - Normalization will convert the data in range between [0, 1]
- Standardization uses the mean and standard deviation to calculate

The following techniques are applied in modeling

- **Year/weekofyear** are applied with **Normalization**
- Other **climate variables** are applied with **standardization**
- NVDIs are normalized data, no need to work on these variables
- Apply these to both train and test sets



Data Processing - Outliers/wrong data ?

Week of year

There are 3 weeks marked as week 53.

There are 52 weeks in a year, let's change the week 53 as week 1 instead.



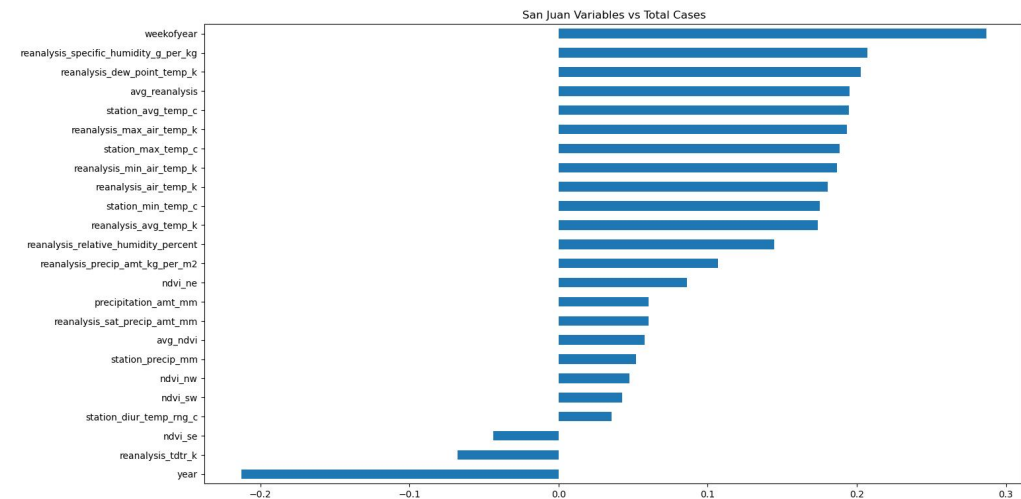
Data Processing - Variable Correlations - SJ data

Total cases is not highly correlated with the variables, however some features are highly correlated to each other.

When correlation > 0.9, suggested to drop columns for San Juan city are:

`'reanalysis_avg_temp_k',`
`'reanalysis_dew_point_temp_k',`
`'reanalysis_max_air_temp_k',`
`'reanalysis_min_air_temp_k',`
`'reanalysis_sat_precip_amt_mm',`
`'reanalysis_specific_humidity_g_per_kg']`

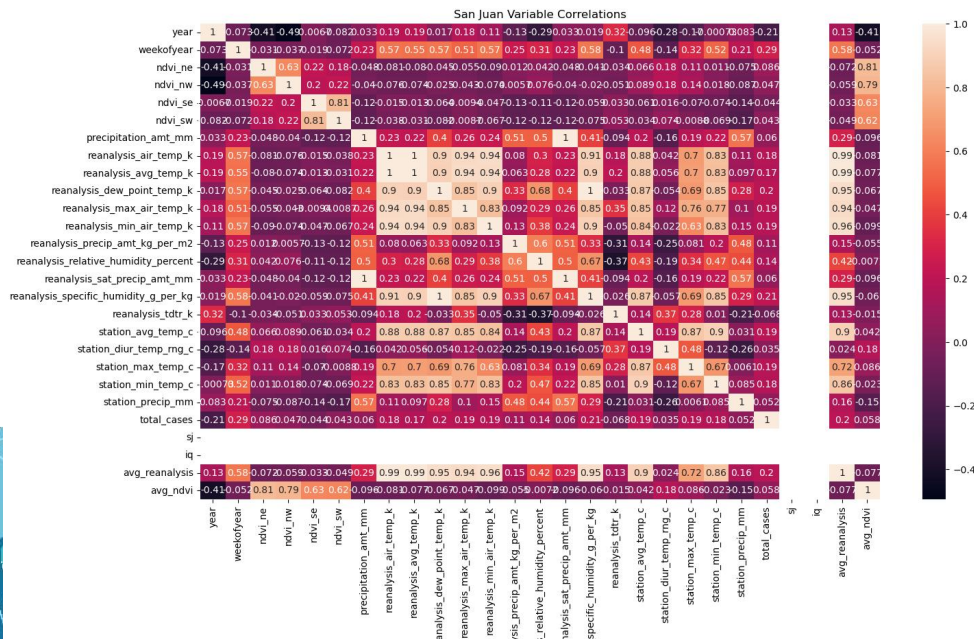
San Juan variables vs total cases



For San Juan city:

After a quick analysis, San Juan cases are more related to weekofyear, reanalysis_specific_humidity_g_per_kg, and more.

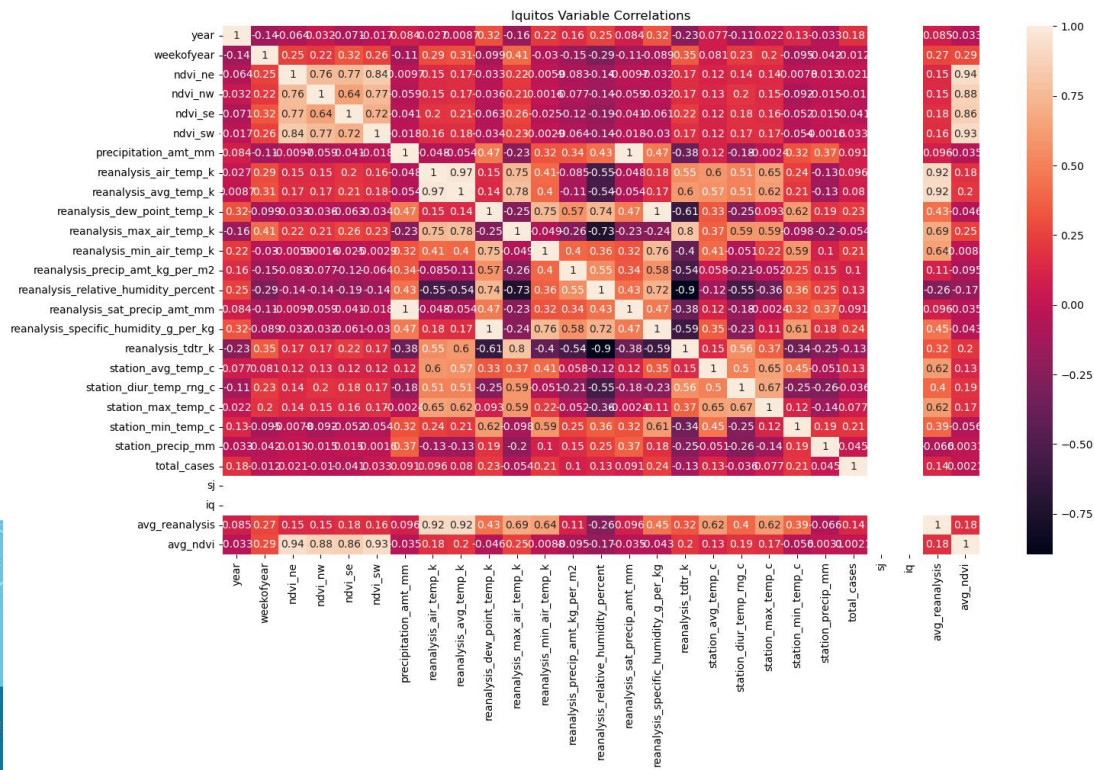
Even reanalysis_specific_humidity_g_per_kg was suggested with high correlation, I will keep this variable for future prediction.



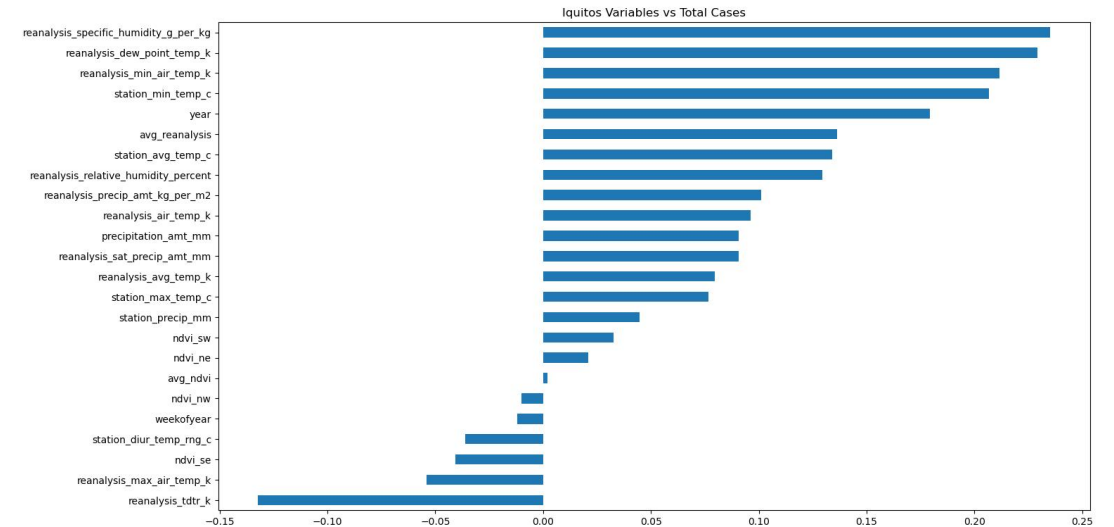
Data Processing - Variable Correlations - IQ data

When correlation > 0.9, suggested to drop columns for Iquitos city are:

`'reanalysis_avg_temp_k',`
`'reanalysis_sat_precip_amt_mm',`
`'reanalysis_specific_humidity_g_per_kg',`
`'avg_reanalysis',`
`'avg_ndvi']`



Iquitos variables vs total cases



For Iquitos:

Total Dengue cases are more related to `reanalysis_specific_humidity_g_per_kg`, `reanalysis_dew_point_temp_k`, and more. One interesting thing is total case has negative value with `weekofyear`, this indicates the data might not have a seasonality.

Even `reanalysis_specific_humidity_g_per_kg` was suggested with high correlation, I will keep this variable for future prediction.

Data Processing insights

From the correlation analysis, we can see:

1. Most highly correlated variables are temperature related, which make sense: **mosquitos prefer to live in areas that are around 70-80 degrees Fahrenheit (21-27C)**. So even though cold weather doesn't kill mosquitoes, they definitely don't like it. At around 60 degrees(15C), they become lethargic and they are incapable of functioning at temperatures below 50 degrees (10C)
2. **Mosquitoes breed in water**, so
 - Wet environment, so higher the humidity higher the total cases
 - Precipitation is also important feature

In summary:

Even reanalysis_specific_humidity_g_per_kg and other temperature variables were suggested with high correlation, but they are important to total dengue cases. I will keep them for future prediction.



Data Processing - Create new features

Temperature values:

When there are same unit, new features can be created by statistical method, such as average of all columns

NVDI index:

Non of the NVDI seems very correlated to total cases, let's create one value average all 4 NVDIs

Month & Season variables

Since the week start date is available, let's create Month and season variables

Cyclical time variables :

Convert week and year into cosine and sine values, cyclical features

```
1 def other_time_features(df):
2     day = 60*60*24
3     week = 7*day
4     year = 365.2425*day
5     df['week_start_date'] = pd.to_datetime(df['week_start_date'])#, format='%Y.%m.%d %H:%M:%S') # convert datatype
6     df['Seconds'] = df['week_start_date'].map(pd.Timestamp.timestamp) # convert to Seconds
7     df['week_sin'] = np.sin(df['Seconds'] * (2 * np.pi / week))
8     df['week_cos'] = np.cos(df['Seconds'] * (2 * np.pi / week))
9     df['year_sin'] = np.sin(df['Seconds'] * (2 * np.pi / year))
10    df['year_cos'] = np.cos(df['Seconds'] * (2 * np.pi / year))
11    df=df.drop(['Seconds', 'week_start_date'], axis=1)
12    return df
```

Lags

From scientific research, we know dengue fever symptoms usually begin 4–10 days after infection and last for 2–7 days. Let's add lagged climate variables in model.

Average weekly cases

For each city we have several year's historical data, let's create average weekly cases as an input variables as well.

Takeaways:

Create new features step is really depend on which model will be used, some of the algorithm doesn't need all these transformation.

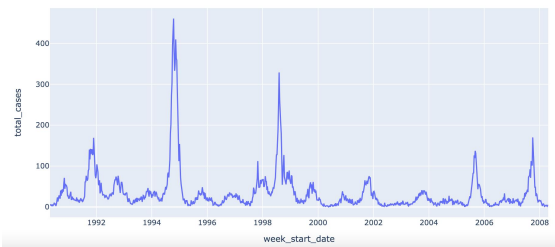
Complex model with more variables will cause overfitting, in each model, pick **important features** for better performance in validation set to **avoid overfitting**.

Data Processing - Split datasets

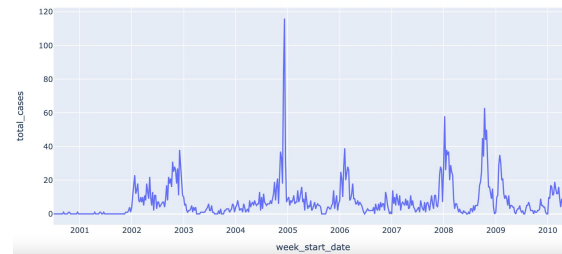
Through our analysis, let's split the data into 2, group by city name.

1. The 2 datasets have **different trends and scales** in terms of total cases of dengue fever.

SJ historical cases

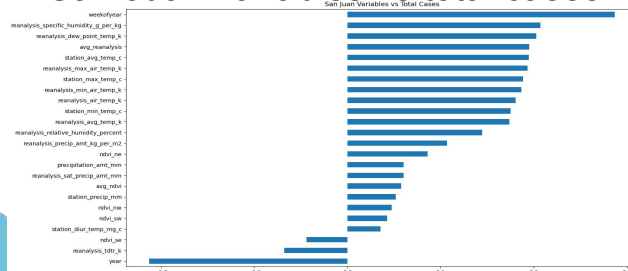


IQ historical cases

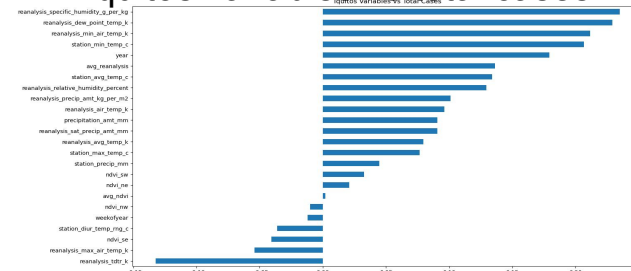


2. Each city's **climate variables contribute differently** to the total cases prediction.

San Juan variables vs total cases



Iquitos variables vs total cases

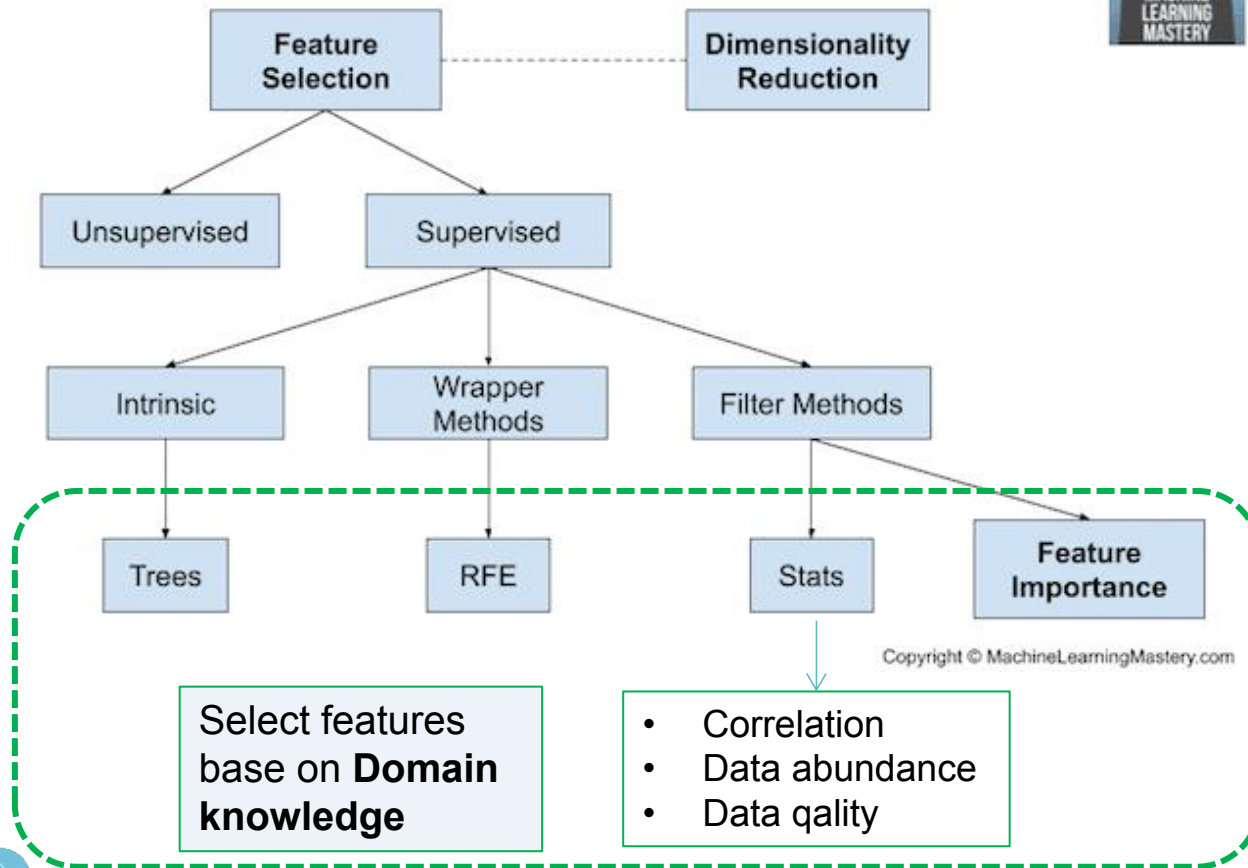


Conclusion:

Split the data by city, and **create 2 separate models** for San Juan and Iquitos predictions prespectively.

Feature Selection

Typical Feature Selection Techniques



Time series forecasting can be framed as a supervised learning problem.

In different models, the feature selection approaches are different, but its all included in the green dashed box.

Some example:

1. *from sklearn, feature_importances_*
 2. *from recursive feature elimination (RFE), RFE(model, n_features_to_select=X)*
 3. *df.corr(method = 'pearson')*
- and more ...

In this project, domain knowledge is very important, so our feature selection part need to be carefully adding the relative features

Step 4: Modeling

Predict the
number of
dengue fever

1 Project Background, what's the question?

2 Exploratory data analysis (EDA)

3 Data Preprocessing

4 Modeling



Model performance

Primary evaluation metric

The metric used for this competition is mean absolute error.

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |f_i - y_i|$$

In conclusion:

After tuning, *xgboost* & *prophet* are good models for the prediction in this exercises.

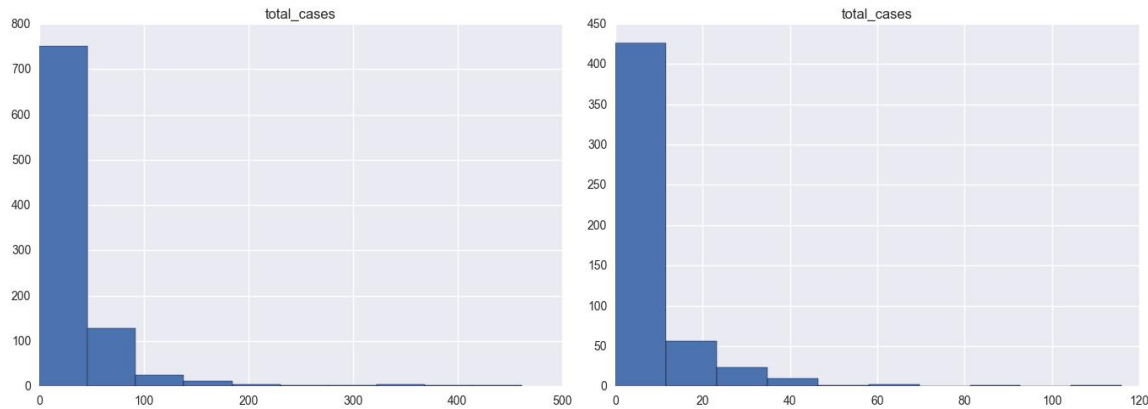
Models Performance Overview

No.	Model	SJ val MAE	IQ val MAE	Submission MAE	python file
1	Benchmark - negativeBionimal	-	-	25.8173	Provided by DrivenData
2	lightGBM	17.598	10.858	32.8774	DengAI_lightgbm_model.ipynb
3	lgb with avg weekly cases	16.095	5.471	32.5938	Data preprocessing & RFR modeling.ipynb
4	Random forest	15.675	5.169	26.0769	DengAI_lightgbm_model_with_weekly_cases.ipynb
5	Prophet with regressor	-	-	26.2476	prophet-trail 1 .ipynb
6	Prophet with tuned seasonality	-	-	25.7981	prophet- 365seasonality.ipynb
7	xgboost	18.508	8.3752	24.1418	DengAI - XGB.ipynb
8	xgboost with cross validation	20.386	4.593	23.8053	DengAI - XGB_cv.ipynb



Mode 1- Benchmark Analysis (Done by DrivenData)

Total cases are all positive values
variance >> mean suggests total_cases can be described by a negative binomial distribution



San Juan
mean: 34.1805555556
var : 2640.04543969

Iquitos
mean: 7.56538461538
var : 115.895523937

Feature selected based on domain knowledge

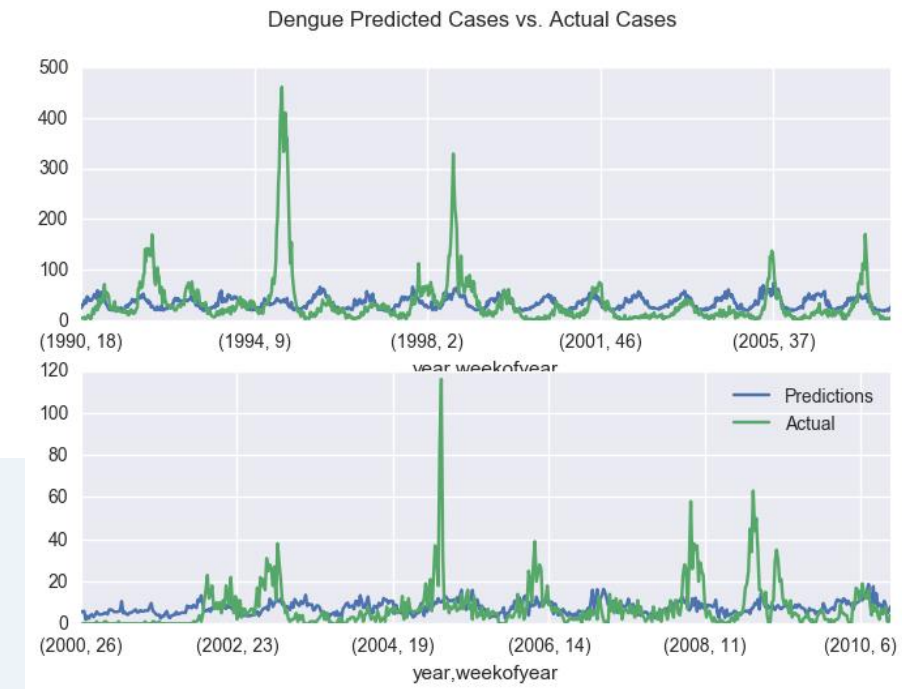
```
model_formula = "total_cases ~ 1 + "\n    "reanalysis_specific_humidity_g_per_kg + "\n    "reanalysis_dew_point_temp_k + "\n    "station_min_temp_c + "\n    "station_avg_temp_c"
```

Reference: <https://drivendata.co/blog/dengue-benchmark/>

Negative Binomial model
Model MAE: 25.8173

Further improvement should be done:

1. Track the seasonality of Dengue cases
2. It missed the spikes that are large outbreaks

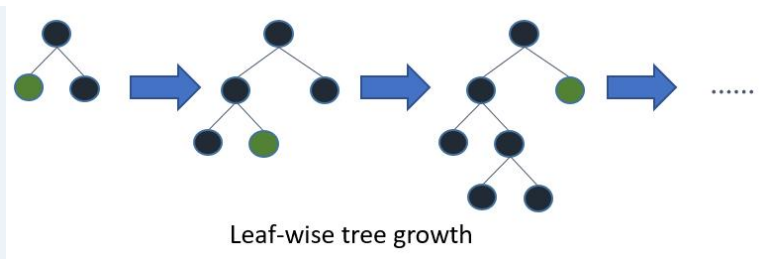


Model 2 - Lightgbm

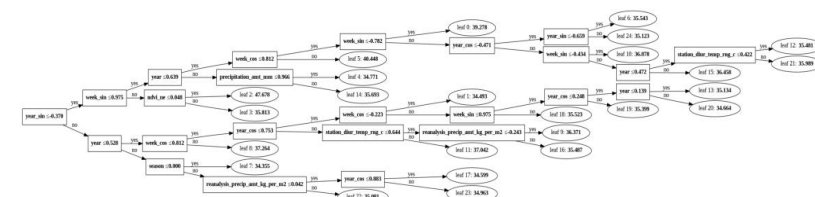
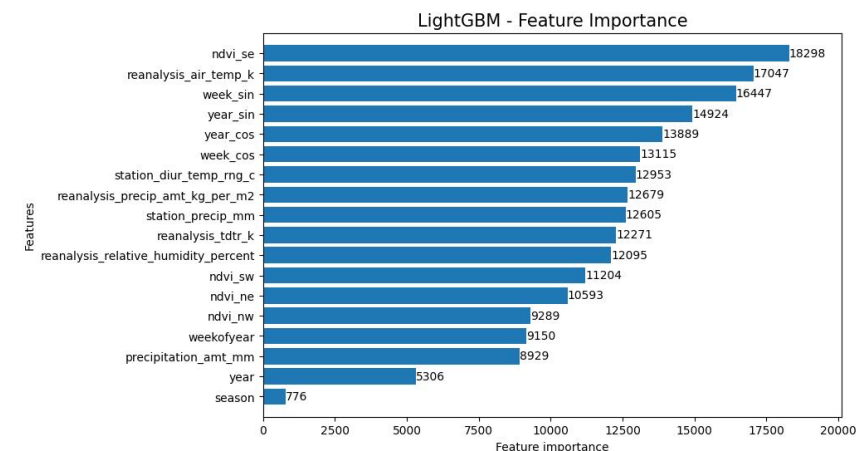
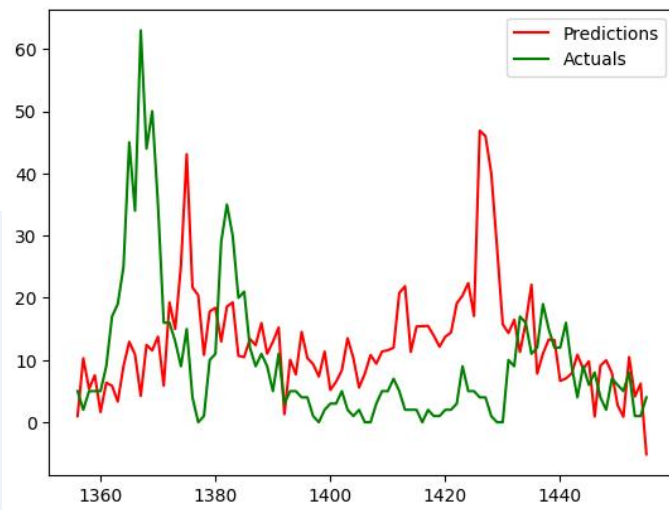
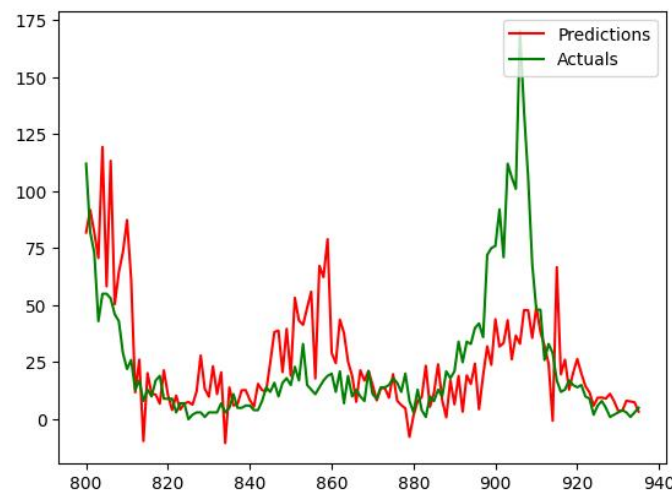
Lightgbm :

A gradient boosting framework that uses tree based learning algorithms. It is designed to be distributed and efficient with the following advantages:

- Faster training speed and higher efficiency.
- Lower memory usage.
- Better accuracy.
- Support of parallel, distributed, and GPU learning.
- Capable of handling large-scale data.



Validation set forecasting:



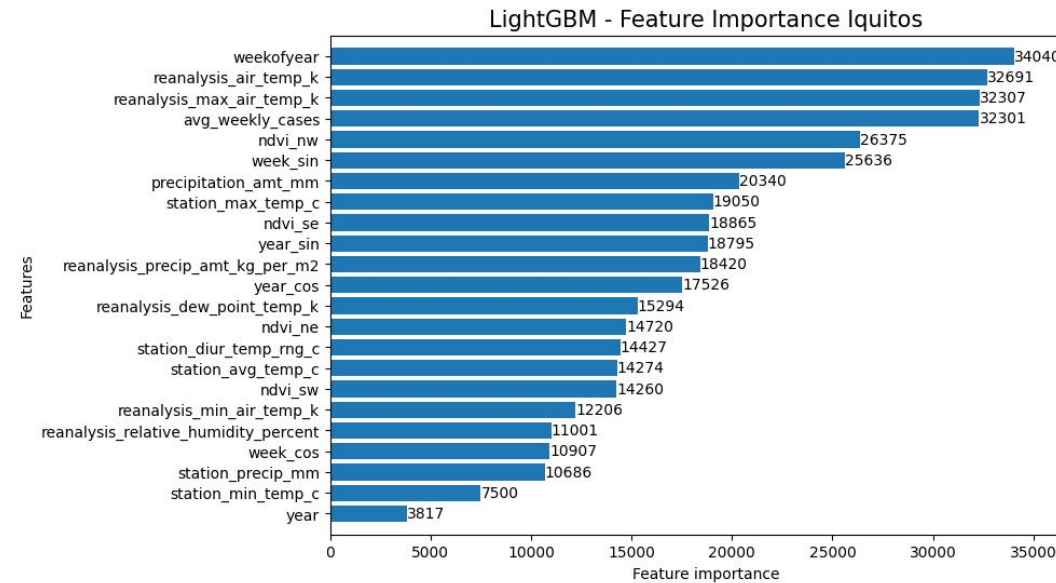
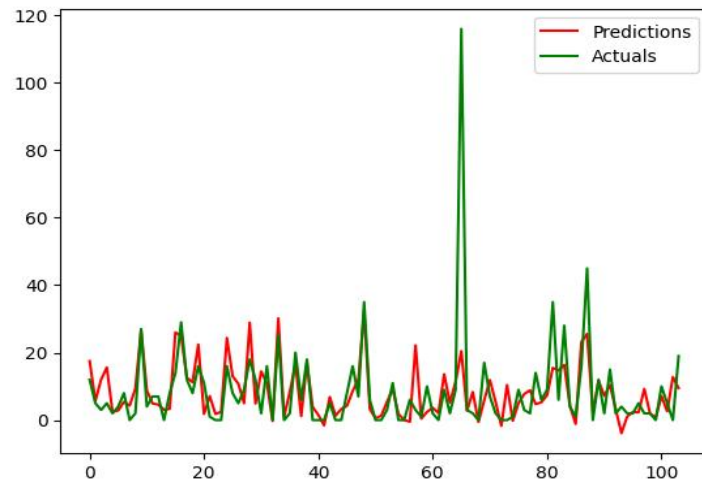
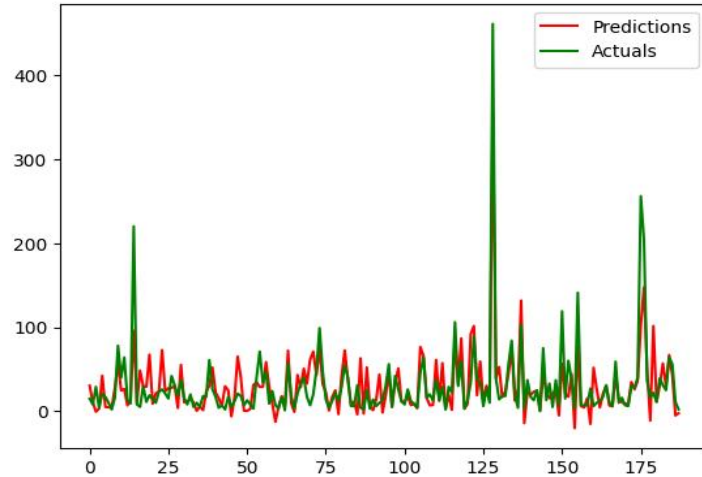
Model MAE: 32.8774

- From the validation set we can see the predictions are not so well
- Model need more work on feature selection



Model 3 - Lightgbm, tuned feature selection

Validation set forecasting:



- Takes very long time to run, need better computational power
I tried to use Google colab

Model MAE: 32.5938

Takeaways:

- Even after tuned the feature selection, MAE was still bad
- Model seems overfitted, since the performance on our testing dataset was not good, but training and validation set has better results.

Future work:

- Need to try to get rid of more features in the future work

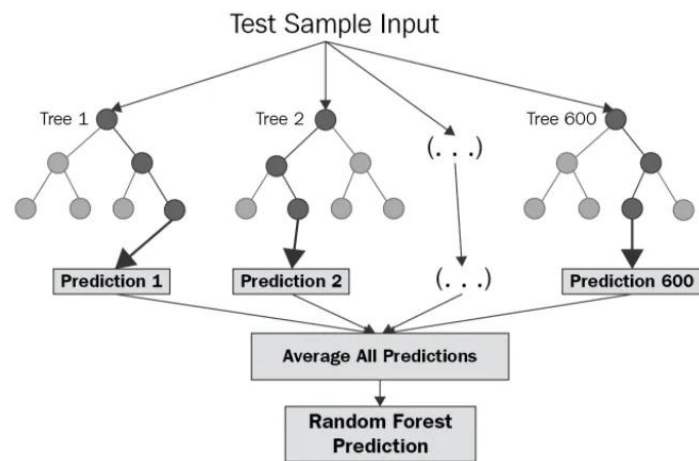


Model 4 - Random Forest Regressor

Random forest:

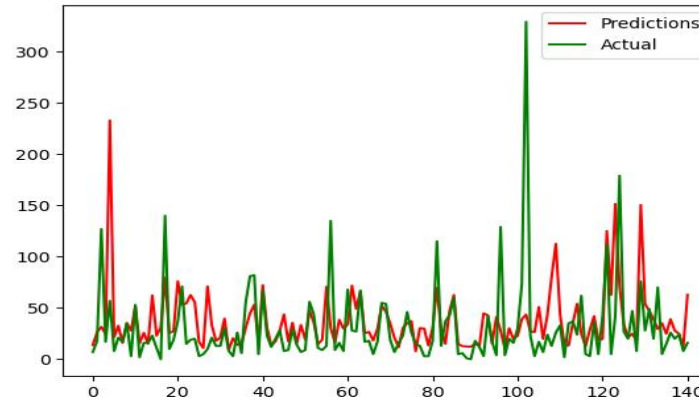
```
rfr = RandomForestRegressor()
```

1. Randomly pick k data points and random features to build the decision tree
2. Make number N of trees and repeat with decision
3. Average all predictions to get the final predicted results

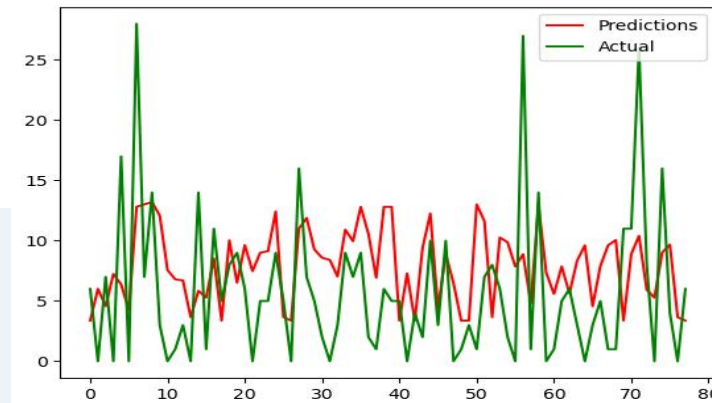


Validation set forecasting:

Dengue Predicted Cases vs. Actual Cases in San Juan



Dengue Predicted Cases vs. Actual Cases in Iquitos



Pros and cons

- Can provide *good accuracy* on prediction
- Can handle both *linear and non-linear relationships* in the data
- Since it is randomly pick features, the prediction can varies from time to time, sometime it will be *hard to explain the feature importance*

Model MAE

Best score

26.0769

Current rank

#2291

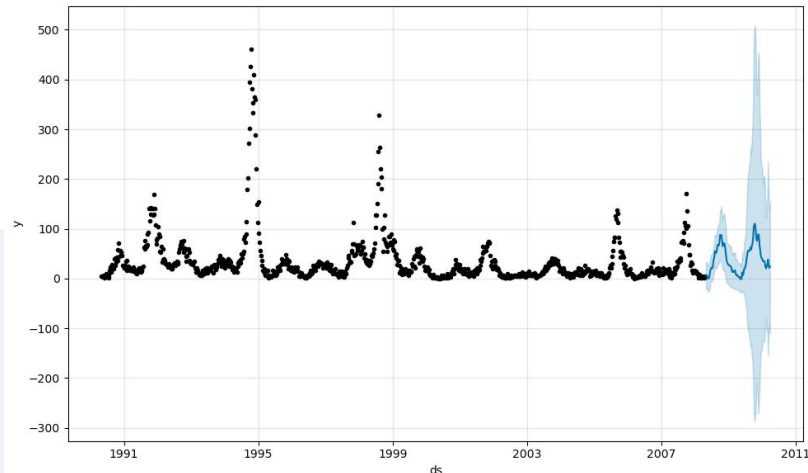


Model 5 & 6 - Facebook Prophet

Prophet

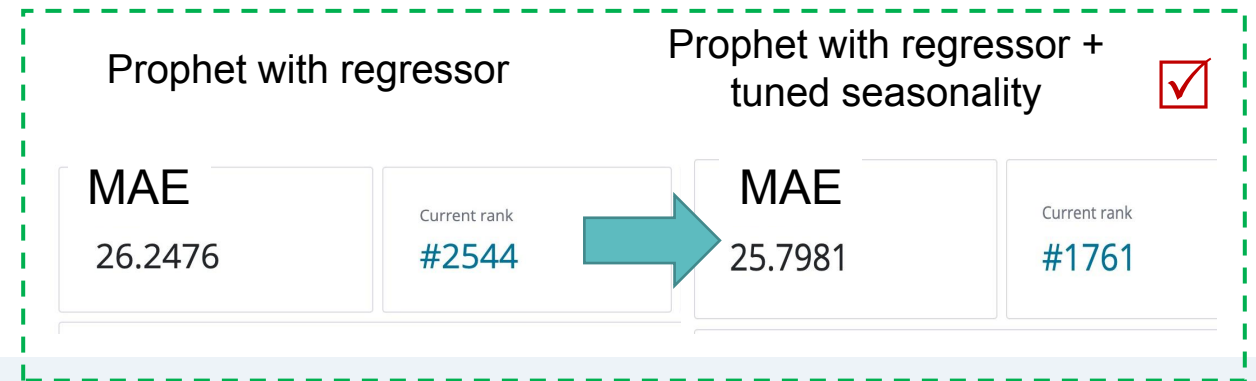
Forecasting time series data based on an **additive model** where non-linear trends are fit with seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data.

Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.



Upon the base Prophet model, there are few things have been done in this exercise:

1. **cross validation**
2. **add_regressor**
3. **add_seasonality**



Reference: https://facebook.github.io/prophet/docs/seasonality,_holiday_effects,_and_regressors.html

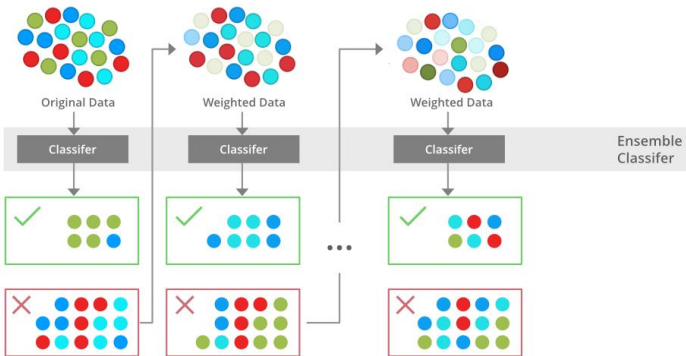


Model 7 - XGBoost

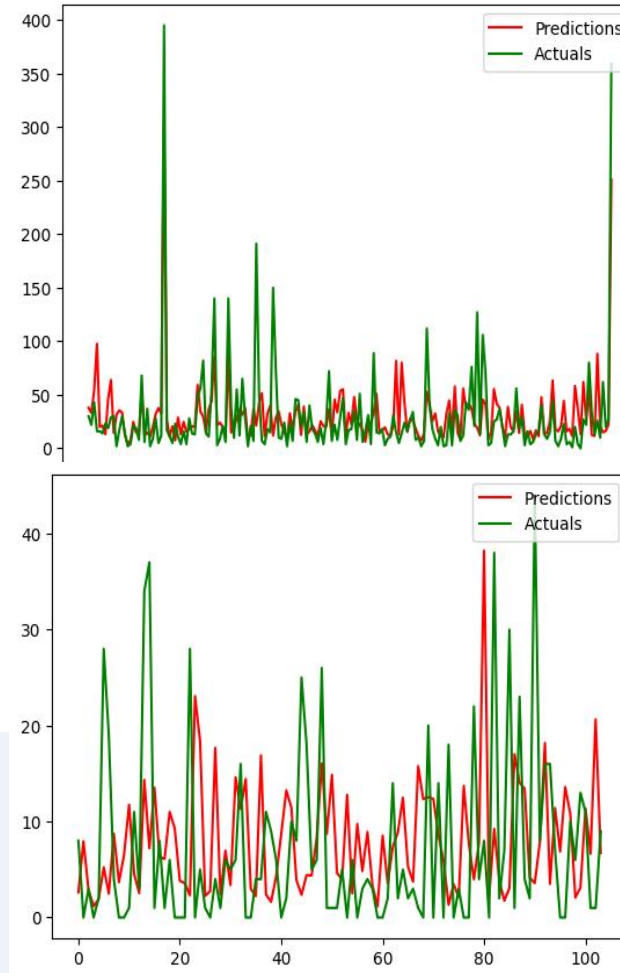
xgboost:

XGBoost stands for Extreme Gradient Boosting. It provides *parallel tree boosting* and is the leading machine learning library for regression, classification, and ranking problems.

The term “gradient boosting” comes from the idea of “boosting” or improving a single weak model by *combining it with a number of other weak models in order to generate a collectively strong model*.



Validation set forecasting:



Pros and cons

- *good accuracy* on prediction
- gives feature importance
- Takes very long time to run, need better computational power
- I tried to use Google colab*

```
[1] from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = 'all'
import os
%matplotlib inline

import pandas as pd
pd.set_option('display.max_colwidth', 50)

import numpy as np
np.bool_ = np.bool_
```

MAE
24.1418

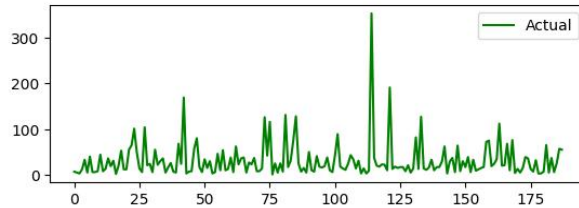
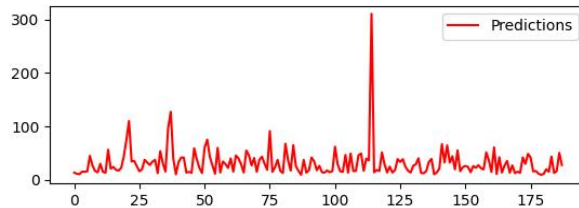
Current rank
#798



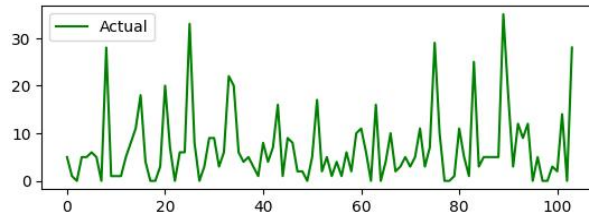
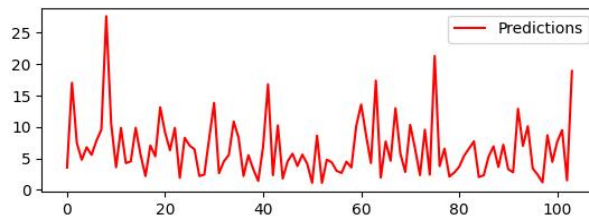
Model 8 - XGBoost with cross validation

Validation set forecasting:

Dengue Predicted Cases vs. Actual Cases in San Juan



Dengue Predicted Cases vs. Actual Cases in Iquitos



xgboost with **cross validation produce better predictions**, but it took long time to run the model on google colab

Tried with different combinations of n_estimators, learning rate, max_depth and more, with cv=3

```
grid_param = {  
    'n_estimators': [500,1000],  
    'learning_rate': [ 0.005, 0.01, 0.1],  
    'max_depth': [4, 8, 12],  
    'subsample': [0.75, 1.0],  
    'colsample_bytree': [0.75, 1.0],  
    'gamma' : [0.025, 0.01],  
    'nthread' : [4,8]  
}
```

Even it took long time, I got good results!

MAE
23.8053

Current rank
#713



Approach to solve this problem

**Predict the
number of
dengue fever**

1 Project Background, what's the question?

2 Exploratory data analysis (EDA)

3 Data Preprocessing

4 Modeling

5 Conclusion and what can be improved?



Project Summary

- From the benchmark analysis, we can see the simple NB model works not bad, with good feature engineering and selections, sometime simpler models can reach the goal.
- Understand the domain knowledge is important, this will definitely help with the feature selection.
- Modeling in python is not the most difficulty part, Features selections are extremely important for the forecasting.
- xgboost model with cross validation works the best so far, there are still a lot of space to improve.

Further improvement



- In the feature engineering part, lag features were added , next step is try to add rolling features
- Recheck the lightgbm regression, why it doesnt perform well, cut the variables to avoid overfitting
- With good computation power, try LSTM or other deep learning methods
- Add additional data, overall pendimic is caused by human movements
- Get into a team, get insights from others

Reference

1. Brownlee, J. (n.d.). Feature Selection with Real and Categorical Data. Machine Learning Mastery. Retrieved from <https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/>
2. Kaleko, D. (n.d.). Feature Engineering: Working with Cyclical Features. David Kaleko's Blog. Retrieved from <https://blog.davidkaleko.com/feature-engineering-cyclical-features.html>
3. DrivenData. (n.d.). DengAI: Predicting Disease Spread. Retrieved from <https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/>
4. World Health Organization. (n.d.). Dengue and Severe Dengue. Fact Sheet. Retrieved from <https://www.who.int/news-room/fact-sheets/detail/dengue-and-severe-dengue>
5. DrivenData. (n.d.). Dengue Benchmark. DrivenData Blog. Retrieved from <https://drivendata.co/blog/dengue-benchmark/>
6. Sharma, A. (n.d.). DengAI: Predicting Disease Spread - STL Forecasting, ARIMA, Box-Jenkins. Medium. Retrieved from <https://medium.com/swlh/dengai-predicting-disease-spread-stl-forecasting-arima-box-jenkins-5f03449179ac>
7. Scikit-learn. (n.d.). RandomForestRegressor - scikit-learn 0.24.2 documentation. Retrieved from <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>
8. LightGBM. (n.d.). LightGBM Documentation. Retrieved from <https://lightgbm.readthedocs.io/en/stable/>
9. Facebook. (n.d.). Seasonality, Holiday Effects, and Regressors. Prophet Documentation. Retrieved from https://facebook.github.io/prophet/docs/seasonality,_holiday_effects,_and_regressors.html
10. GitConnected. (n.d.). Random Forest Regression. Retrieved from <https://levelup.gitconnected.com/random-forest-regression-209c0f354c84>
11. RoboticsBiz. (n.d.). Pros and Cons of Random Forest Algorithm. Retrieved from <https://roboticsbiz.com/pros-and-cons-of-random-forest-algorithm/>
12. Hyndman, R. J., & Athanasopoulos, G. (n.d.). Chapter 8: Stationarity. Forecasting: Principles and Practice. Retrieved from <https://otexts.com/fpp2/stationarity.html>
13. NVIDIA. (n.d.). XGBoost. Retrieved from <https://www.nvidia.com/en-us/glossary/data-science/xgboost/>
14. GeeksforGeeks. (n.d.). XGBoost. Retrieved from <https://www.geeksforgeeks.org/xgboost/>












Thank you!

Any Questions and suggestions?



Appendix

Submission Scores

Public score	Who	Details
26.5793	 szhang68	id-241483 · 2d 11h ago · NegativeBinomial
26.2476	 szhang68	id-241485 · 2d 10h ago · prophet1
26.3029	 szhang68	id-241488 · 2d 6h ago · tuned prophet
26.0769	 szhang68	id-241491 · 2d 1h ago · random forest
25.7981	 szhang68	id-241497 · 1d 20h ago · prophet 365 seasonality
27.0216	 szhang68	id-241511 · 1d 8h ago · random forest ver2
24.1418	 szhang68	id-241520 · 1d 2h ago · xgb
32.5938	 szhang68	id-241523 · 1d 1h ago · lgb with avg weekly cases variable
23.8053	 szhang68	id-241554 · 0min ago · xgb with cross validation

Final Ranks

712 out of 13746 by Sep 15,2023

HEALTH

DengAI: Predicting Disease Spread

Using environmental data collected by U.S. Federal Government agencies, can you predict the number of dengue fever cases reported each week in San Juan, Puerto Rico and Iquitos, Peru? Competition hosted by DrivenData.



Intermediate practice



3 months left



13,746 joined



Navigation

- Home
- About
- Problem description
- Official rules
- Leaderboard
- Discussion (30)
- Data download
- Submissions (10)**

Submissions

- To help you track your progress during the competition, each submission is scored against publicly available test data to give a "public score".
- The primary evaluation metric is Mean Absolute Error. [Show more.](#)

Best score

23.8053

Current rank

#712

Submissions used

2 of 3