Group 30

DengAI Competition

Final Report

Abeygunawardhana H. A. W. (140004E);
Jayatilake S. A. P. (140263U);
Perera P. D. I. T. S. K. (140462E);
Thewa Hettige S. P. (140623B)

7-20-2018

CONTENTS

Introduction 3
Problem Statement 3
Objectives 4
Methodology 4
Dataset 4
Data analysis 7
Time series analysis 14
Data preprocessing 16
Data cleaning 16
Data transformation 17
Data reduction 17
Model selection 18
Manual Model Selection 18
Automated model selection 18
Best model that worked 19
Training and evaluation 19
Deep Neural Networks based model 19
Random Forest 20
Boosted trees/AdaBoost 21
Boosted Trees with STL 21
Dual-Linear Regression 21
Parameter tuning 21
Prediction 22
Discussion 22
Conclusion 24
References 24

INTRODUCTION

Dengue fever is a deadly viral disease transmitted by mosquitos. Upon infection, the patients would show fever, rash, and muscle and joint pain. In severe cases it could cause severe bleeding, low blood pressure, and even death. Since the spreading of dengue fever is done by mosquitos, many studies show that spreading of this disease is positively correlated with climate factors such as temperature, humidity and precipitation amount. With having enough historical data related to the dengue disease and the climate changes, it invites the question of whether it is possible to use the correlation of the data to predict an epidemic before it happens.

Although it is possible to study the data in conventional statistical methods, there is an emerging trend of using data science for analyzing and exploring large sets of data related to the area of disease forecasting. It is very efficient to use data science for analyzing correlations among data sets because the available historical data could be having noise, or the data could be incomplete. So, to come up with accurate predictions, it is important to prepare the data before it's being used for the prediction purposes.

DengAI is a competition hosted by DrivenData Foundation with the purpose of combining the above two concepts to create a global community of researchers and data science enthusiasts to solve a common issue which affects all mankind. This report summarizes the efforts done by authors to create an accurate prediction model for the above competition with the intention of exploring the possibilities of data science and by doing so, be of service to the mankind.

PROBLEM STATEMENT

In this project, we address the problem of predicting the total number of patients who are infected with dengue in a given week of the year, using climate data such as weather station measurements, satellite precipitation measurements, NOAA's NCEP Climate Forecast System Reanalysis measurements and Satellite vegetation index information.

OBJECTIVES

The data science competition "DengAI: Predicting Disease Spread" hosted by DrivenData is about predicting the number of dengue cases each week (in each location) based on environmental variables describing changes in temperature, precipitation, vegetation, and more [1].

The goal of this project is to research and build a model for the above mentioned "DengAI" competition which can predict the total_cases (i.e.: total number of dengue cases) label for each (city, year, weekofyear) in the test set provided by them.

METHODOLOGY

Although followed these steps to achieve a good accuracy in predicting the number of dengue cases, sometimes these steps are interleaved. That is because visualizing and analyzing steps always come in the middle of preprocessing or parameter tuning.

1. Gathering data
2. Perform exploratory data analysis
3. Data preprocessing
4. Choosing a model
5. Training and evaluation
6. Parameter tuning
7. Prediction

Dataset

The dataset consists of climate data and the total number of reported dengue patients in a weekly timescale. The data is available in following categories.

City and date indicators

This data describes the city that's related to the data, and the starting date of the week of the data. Following attributes are available in the dataset from category of data.

- city – City abbreviations: ‘sj’ for San Juan and ‘iq’ for Iquitos
- week_start_date – Date given in yyyy-mm-dd format

NOAA's GHCN daily climate data weather station measurements

Global Historical Climatology Network (DHCN) is an integrated database of daily climate reports from land surface stations across the globe. This data is subjected to a common suite of quality assurance reviews and contains records from over 100,000 stations in 180 countries and territories. Following attributes are available in the dataset from category of data.

- station_max_temp_c – Maximum temperature
- station_min_temp_c – Minimum temperature
- station_avg_temp_c – Average temperature
- station_precip_mm – Total precipitation
- station_diur_temp_rng_c – Diurnal temperature range

PERSIANN satellite precipitation measurements (0.25x0.25-degree scale)

The Precipitation Estimation from Remotely Sensed Information using Artificial Neural Networks- Climate Data Record (PERSIANN-CDR) is a daily rainfall estimate at a spatial resolution of 0.25 degrees in the latitude band 60S - 60N from 1983 to the near-present. Following attributes are available in the dataset from category of data.

- precipitation_amt_mm – Total precipitation

NOAA's NCEP Climate Forecast System Reanalysis measurements (0.5x0.5-degree scale)

The National Centers for Environmental Prediction (NCEP) provides reanalysis data which will represent data about how weather and climate are changing over the time. This type of reanalysis data is used for comparing current climate conditions with the past climate conditions. Following attributes are available in the dataset from category of data.

- reanalysis_sat_precip_amt_mm – Total precipitation
- reanalysis_dew_point_temp_k – Mean dew point temperature
- reanalysis_air_temp_k – Mean air temperature
- reanalysis_relative_humidity_percent – Mean relative humidity
- reanalysis_specific_humidity_g_per_kg – Mean specific humidity
- reanalysis_precip_amt_kg_per_m2 – Total precipitation
- 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

Satellite vegetation - Normalized difference vegetation index (NDVI) - NOAA's CDR Normalized Difference Vegetation Index (0.5x0.5-degree scale) measurements

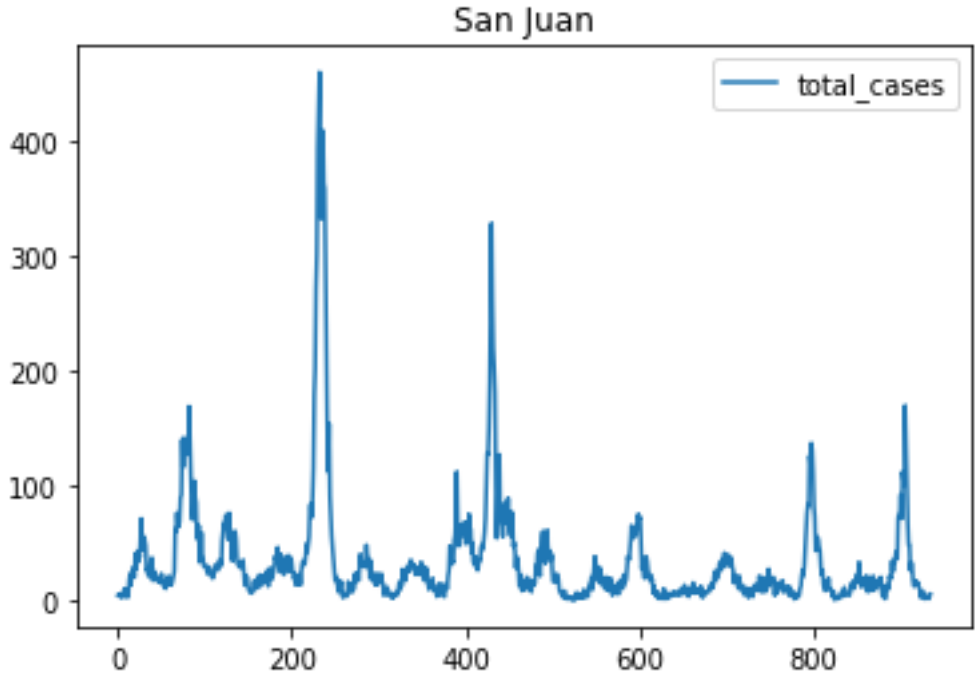
This data represents a measure of vegetation in a given area, by measuring the difference between near-infrared which vegetation strongly reflects and red light which vegetation absorbs. So, these values can be used to determine the availability of water and vegetation in an area over time. Following attributes are available in the dataset from category of data.

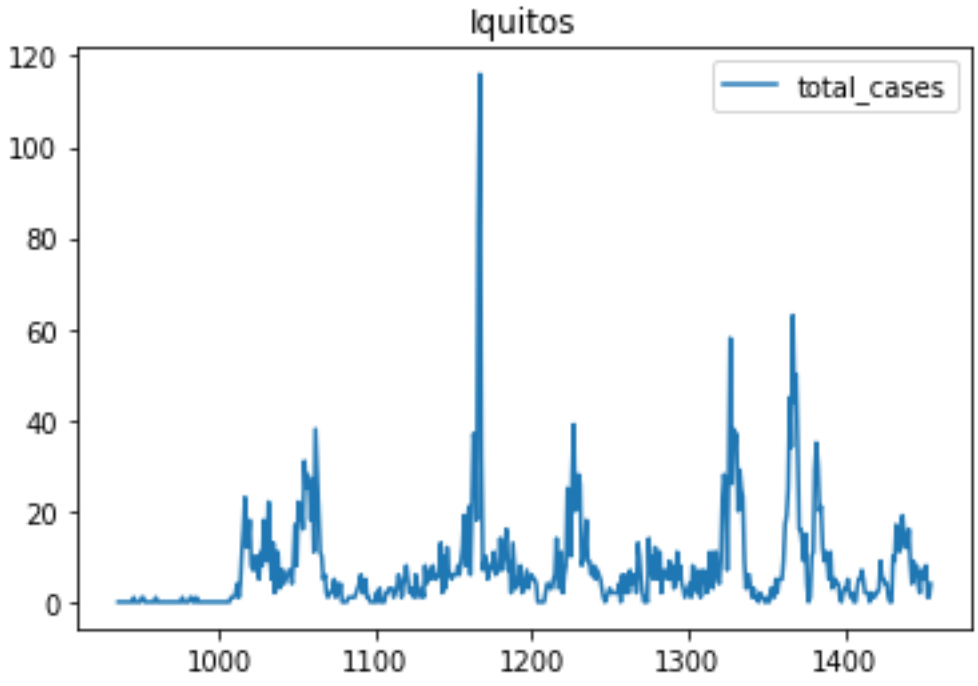
- ndvi_se – Pixel southeast of city centroid
- ndvi_sw – Pixel southwest of city centroid
- ndvi_ne – Pixel northeast of city centroid
- ndvi_nw – Pixel northwest of city centroid

Data analysis

For analyzing the feature set, we combined the features both from training and test data sets.

First look at the data revealed that the two cities Iquitos and San Juan has somewhat different properties. Most importantly, the total cases (target variable) contained very different data. This could be because of the population (density) of the towns. Therefore, we decided to separate the data based on cities. (First few submissions using neural networks without separating cities proves this point.)

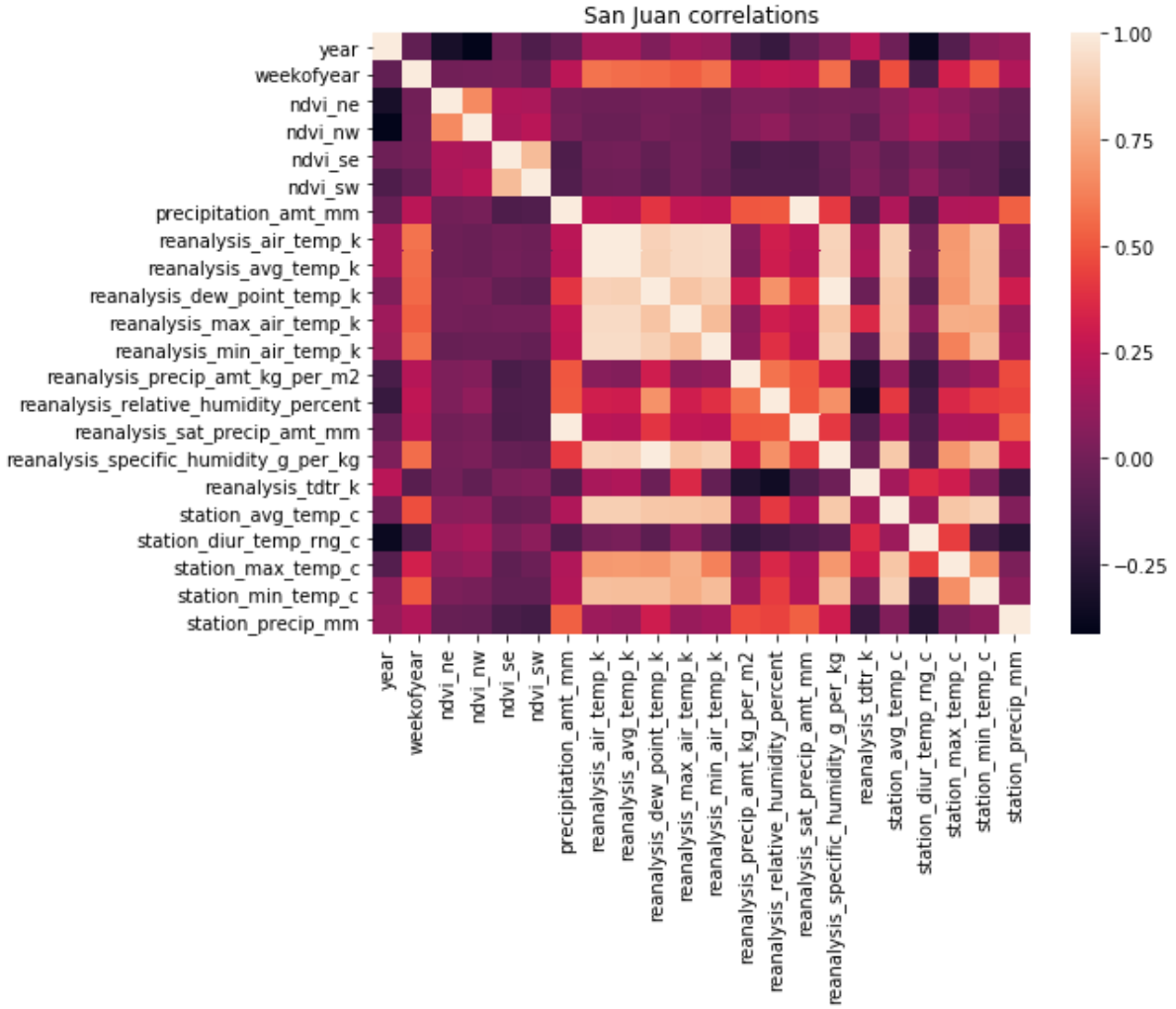


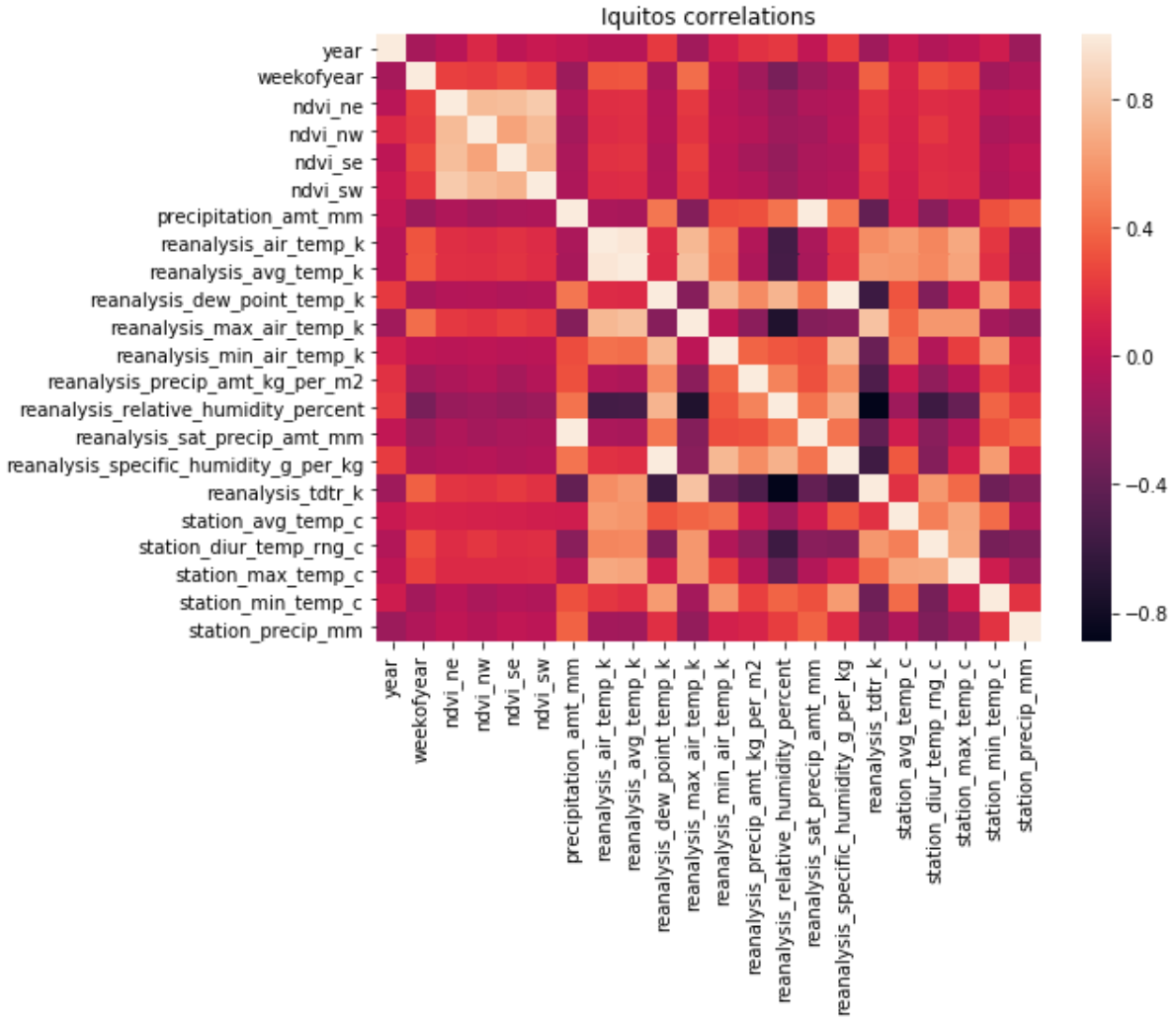


We tested the cross-correlation among features. This is to identify repeated features and identify features that can be reduced. The following correlation heatmaps use Pearson's correlation coefficients.

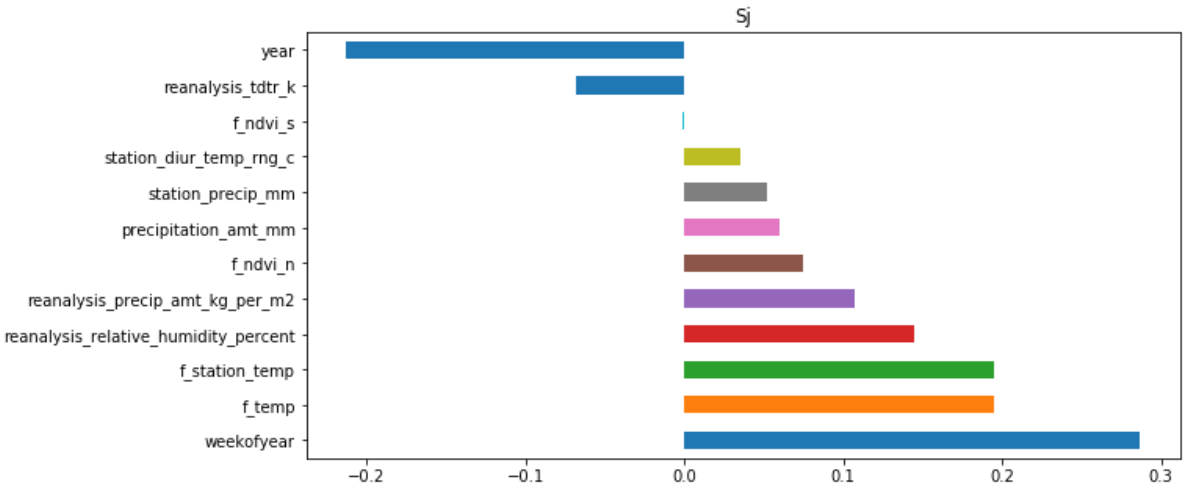
When we consider a 95% correlation, the features 'reanalysis_avg_temp_k', 'reanalysis_sat_precip_amt_mm', 'reanalysis_specific_humidity_g_per_kg' can be eliminated since they have very similar features left in the dataset.

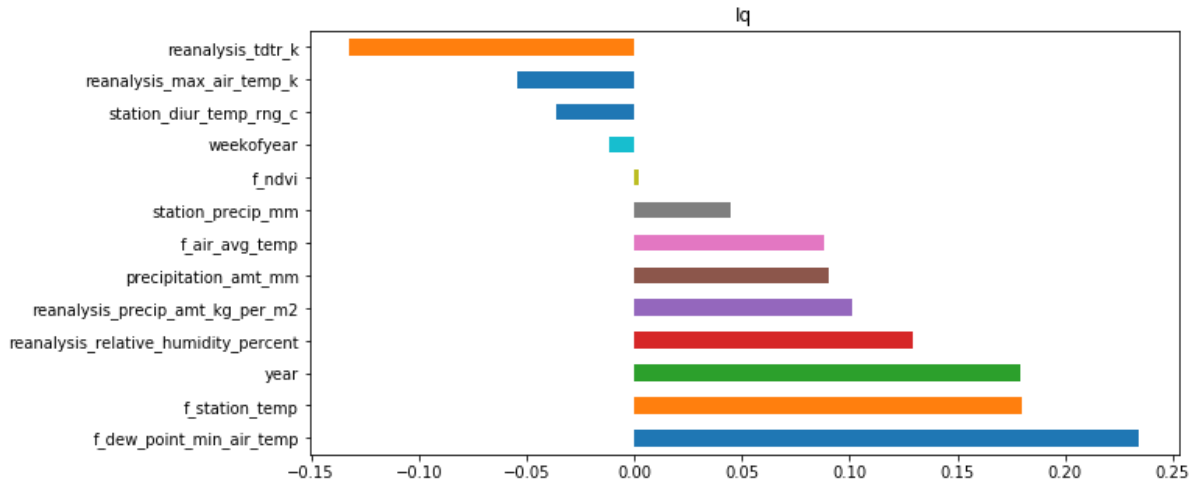
In San Juan, we can notice that the temperature measures have a good correlation whereas in Iquitos, NDVIs and humidity shows a good correlation.





As far as the correlation with the target variable ‘total_cases’ is concerned, there was no obvious correlation to be found. (After eliminating/combining similar features.)

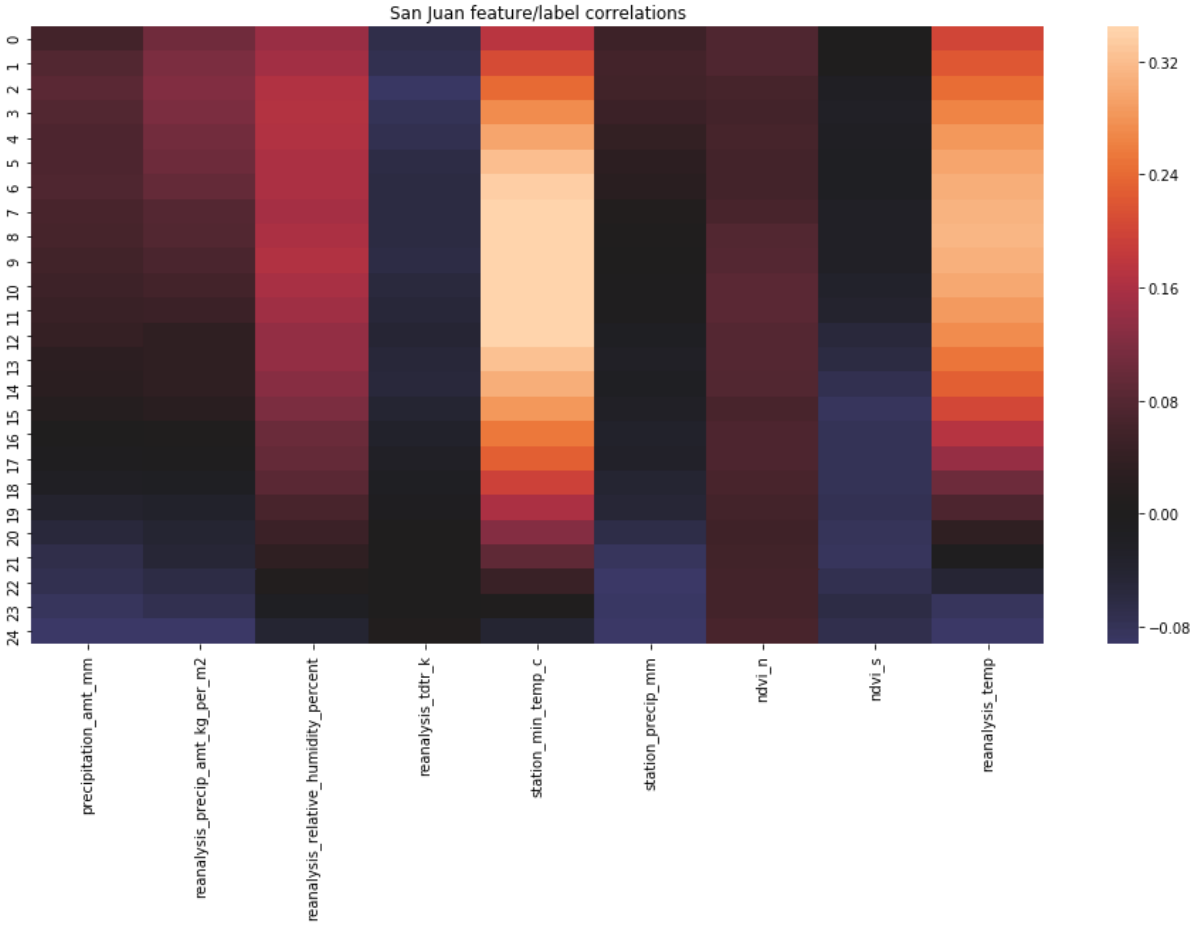


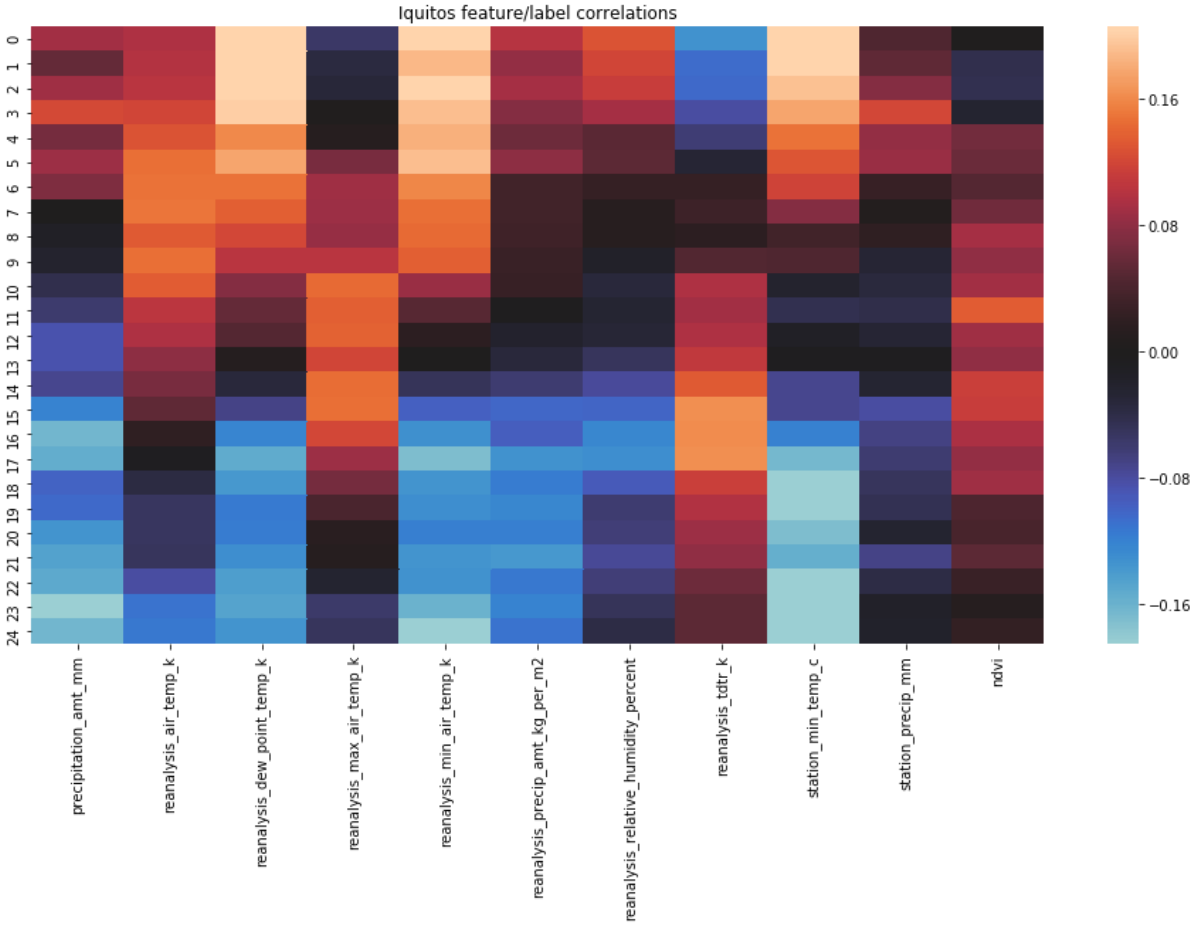


Doing more background research, it was clear that features from weeks other than the current week can have a significant impact on total cases. This is because:

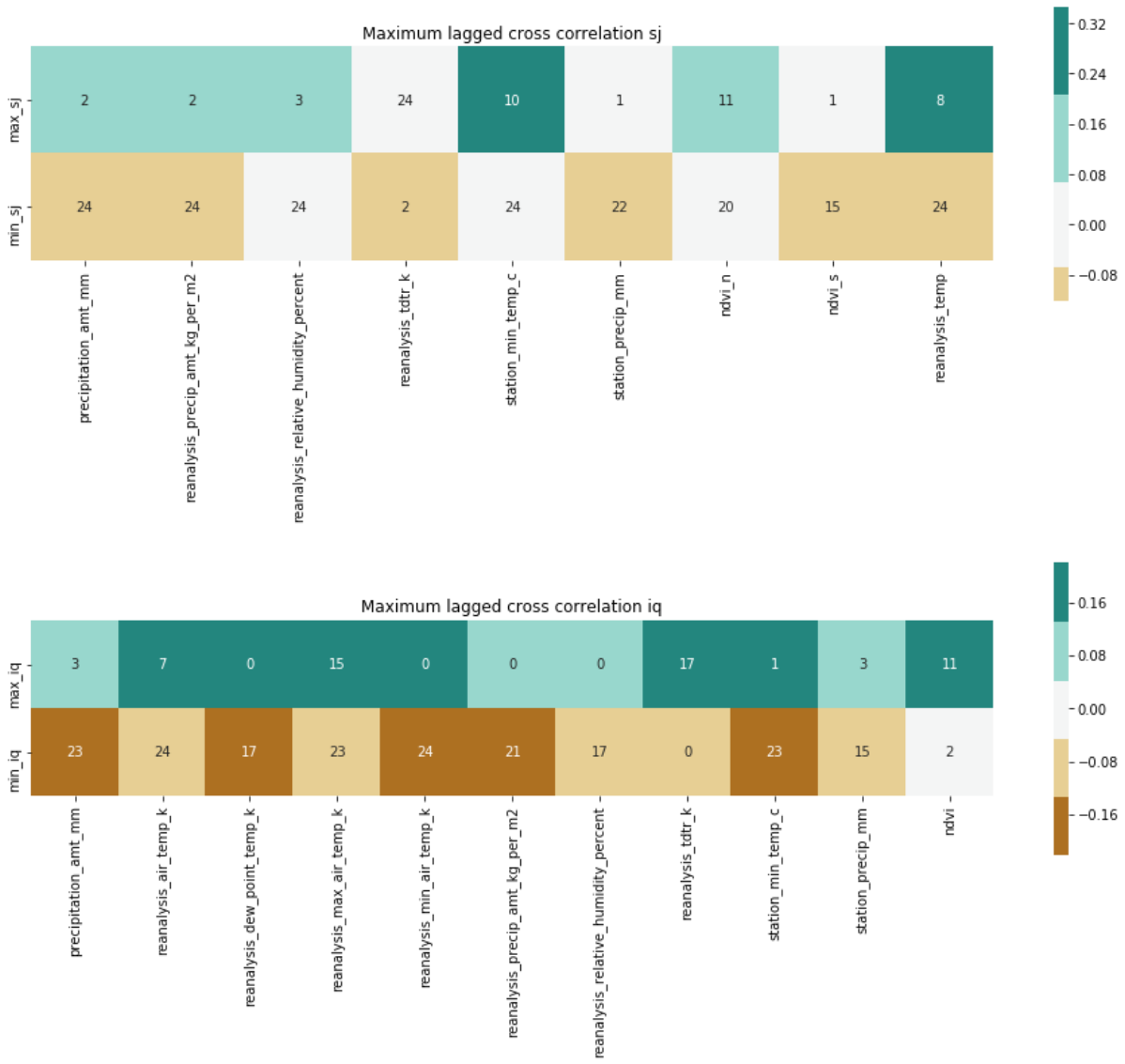
- Mosquito life cycle is 8 to 10 days (This also depends on temperature values). So, from egg to an adult mosquito should at least take 8 days.
- The incubation of dengue virus inside the mosquito is 8 to 12 days, which means, if a mosquito to get the dengue virus, it takes 8 to 12 days for the mosquito to be able to transfer it to a human.
- The incubation period of dengue virus inside human body is typically 2 weeks, so for a dengue-infected human to be able to transfer the disease to a mosquito it will take 2 weeks.
- An adult mosquito lives up to 4 weeks.
- It takes few days to show the symptoms and get diagnosed with dengue.

We tried lagging the feature data and visualizing the correlation with each feature and total cases vs the lag.


San Juan feature/label correlations
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
precipitation_amt_mm
reanalysis_precip_amt_kg_per_m2
reanalysis_relative_humidity_percent
reanalysis_tdtr_k
station_min_temp_c
station_precip_mm
ndvi_n
ndvi_s
reanalysis_temp
0.32
0.24
0.16
0.08
0.00
−0.08



Following are heatmaps of correlations of features with total cases for the 2 cities when features are lagged by up to 24 weeks. It also considers the negative correlation. What is inside each square is the weeks lagged.



Correlations are improved but only slightly. Although weeks lagged up to 24 are tested, looking at the domain knowledge we can assume that a lag of more than 4 months should correspond only to some seasonal pattern, not the impact a feature making its direct consequences after that many weeks.

What can be noted is that station temperatures and some reanalysis temperatures have a consistent correlation with total cases (not very clear in Iquitos dataset). All reanalysis_precip_amt_kg_per_m2, reanalysis_relative_humidity_percent and station temperatures are listed as highly correlated features in both cities. Although there are other features with greater correlations, they are not consistent across datasets. For the disease dengue, since the viruses and Aedes mosquitos have mostly similar behaviors across countries, the factors for the disease should be similar.

TIME SERIES ANALYSIS

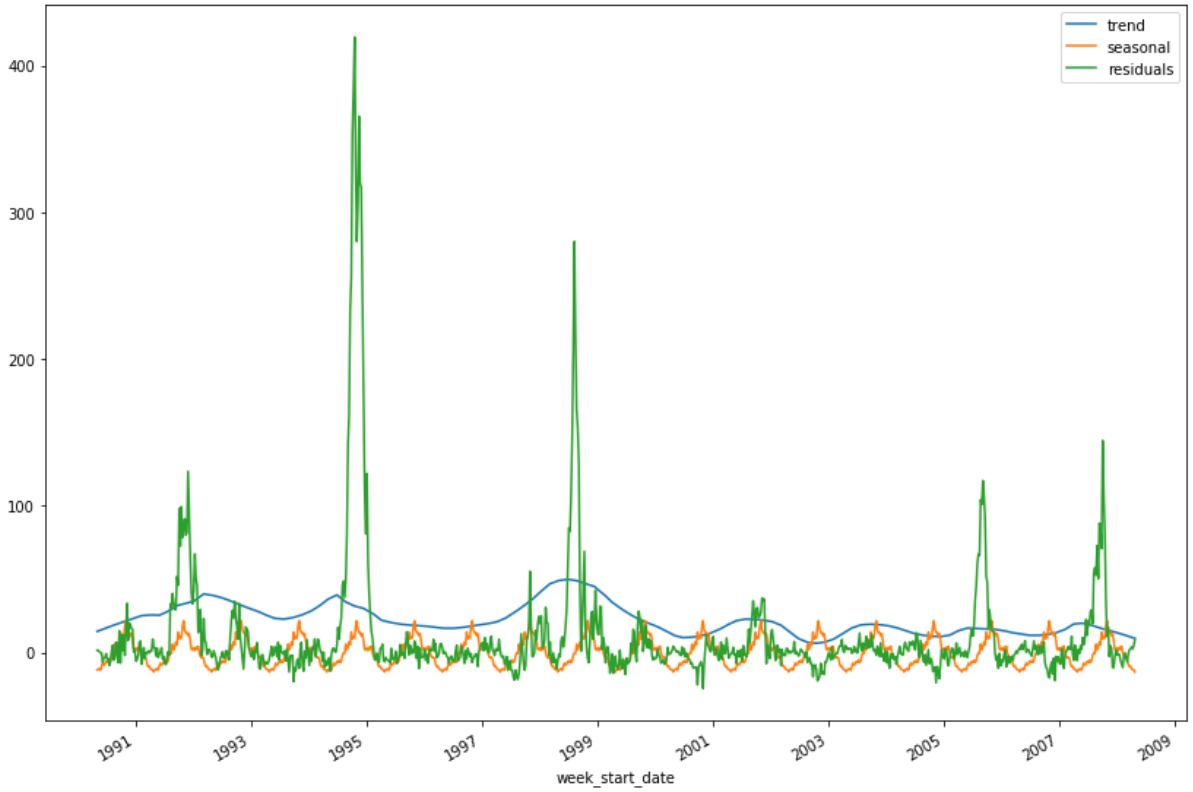
Since most of the features seemed to be strongly affected by the week of year, we tried a statistical time series analysis to separate seasonal, trend and residual components. Time series have several components:

- Trend component: affects the long-term progression of the series. This is typically a persistent increasing or decreasing of the value. For example, global warming may cause the temperature to increase gradually, when many years is considered.
- Cyclical component: Reflects repeated, but non-periodic fluctuations. The duration is usually more than two years.
- Seasonal component: This reflects seasonality of the series, which is also cyclic but occurs over a fixed period. For example, temperature variation over seasons, or the rain is a seasonal variation.
- Irregular component: This represents the residuals or remainder of the time series after the other components have been removed. This is often called as the noise.

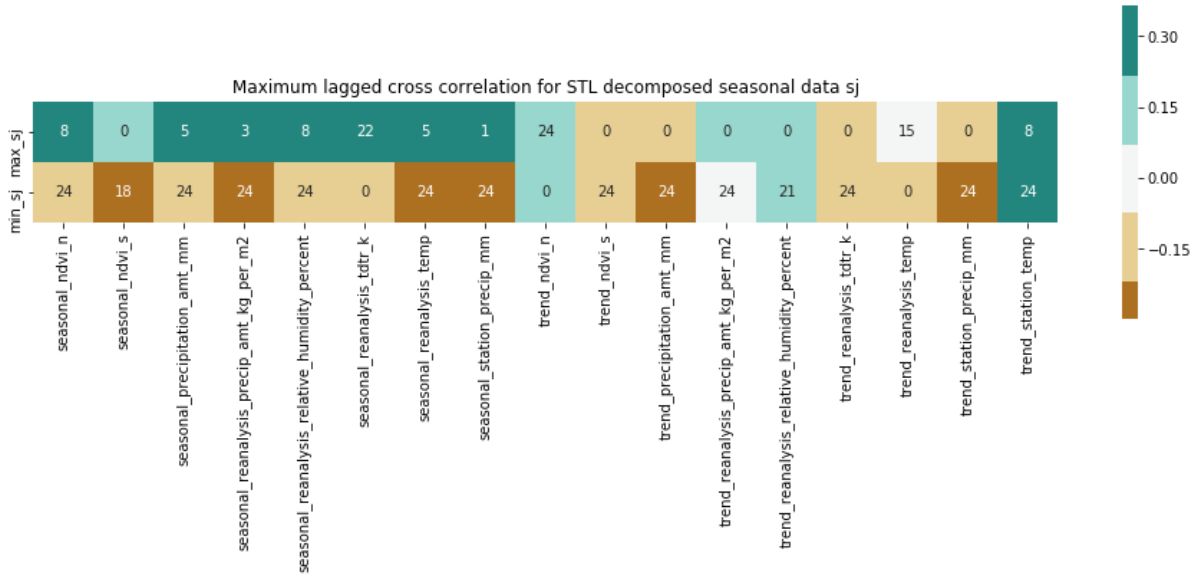
The composition of the components can be either additive or multiplicative. Since no exponential variation is visible, we assumed the composition to be additive.

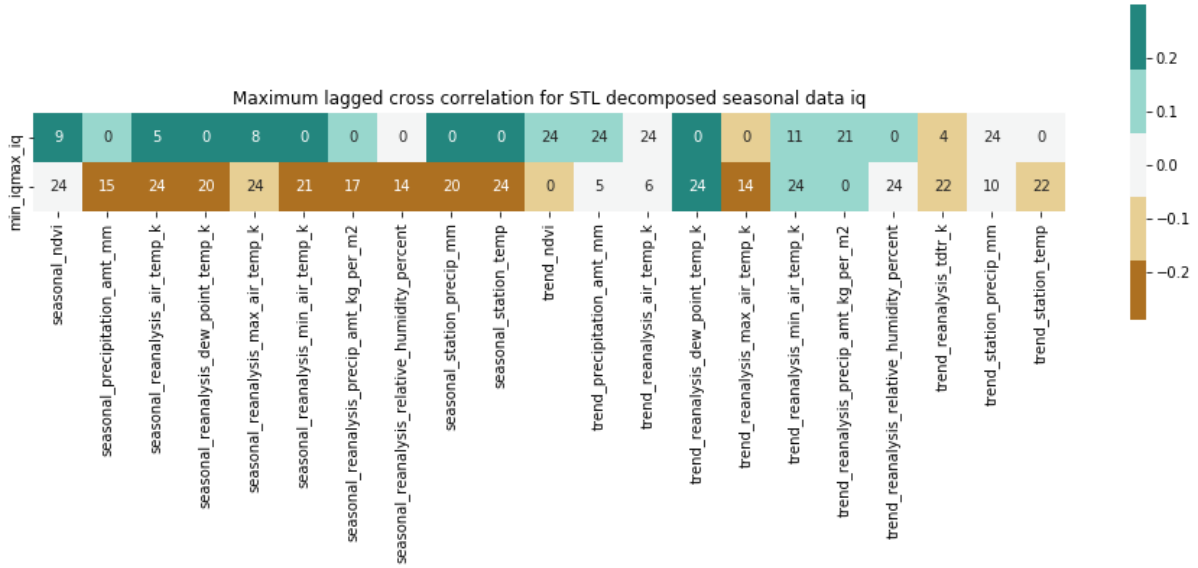
The python-native statsmodels library documentation states its seasonal_decompose function as “naïve” [2]. So, we borrowed R’s “robust” STL (Seasonal and Trend decomposition using Loess) decomposition function to python using r2py bridge.

Example decomposition of total cases (sj):



Like the above described, we tried multiple levels of lagging on each component to find out which correlated with total cases the most.





The correlation did not increase much as expected after the decomposition. Component decomposition of total cases and matching them with corresponding components did neither show a great change. Also, residuals appeared with higher variations than seasonal and trend components. This might suggest residuals are not just noise.

However, we tried using results from STL decomposition in a few models that we constructed.

Data preprocessing

DATA CLEANING

Imputation of missing values

In San Juan dataset, most missing values were NDVIs while in the Iquitos dataset most missing values were station temperatures.

There are several methods for imputation of missing values. Filling them with mean, median or mode, carrying forward the previous value, bringing backward the next value, and interpolation is some of them. Although statistical central tendency measures (mean etc.) look like a good choice at the first sight, they tend to decrease variation and do not yield very good results. Forward fill and backfill are better at preserving patterns. For large ranges of missing data, interpolation seem to perform better. Therefore, linear interpolation is used to impute missing data.

Imputing values using ‘weekofyear’ seemed like a smart idea at first but the trend of each feature (when time series is considered) was against this decision.

Identify outliers and/or smooth

Moving averages were used most of the time to smooth-out outliers. Although median is robust at removing outliers, it removes a significant amount of important “spikes” in data, reducing the impact. Also, the graphs of median-smoothed data look jagged. As the moving window, different amounts were used for different purposes and models, but most commonly, a window of 5 weeks was used.

DATA TRANSFORMATION

Normalization

Except temperatures, other features had different units. For the station temperature measurements, the unit was Celsius while for reanalysis temperature measurements it was Kelvin. All values were converted to Celsius to avoid confusion. Diurnal temperature range is a temperature difference although it was specified in kelvin. Therefore, it was kept as is.

Since neural network models required normalization, Z-score normalization was used.

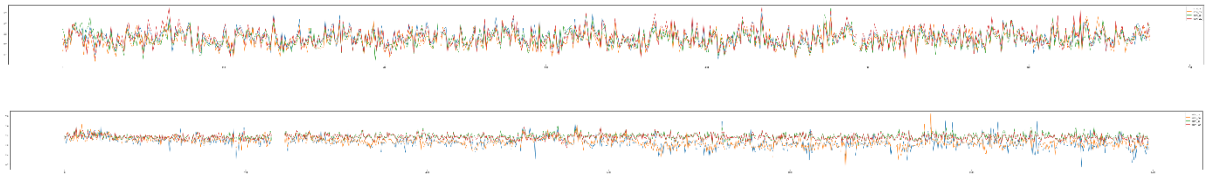
Attribute construction

For some models, we used STL decomposed trend and seasonality components skipping residuals.

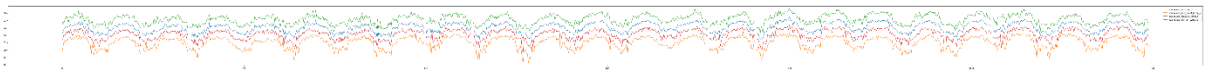
DATA REDUCTION

Reducing the number of attributes

All NDVI values in Iquitos were combined into one using mean. Since NDVI in two different directions was noticeably different, we could reduce them to two components instead.



For San Juan, all reanalysis temperatures showed the same visual pattern. They were combined using mean to reduce the number of features.



The station diurnal temperature range was redundant in both cities. Thus, it was dropped.

Other station temperature variances for both cities looked similar. Specially, the average temperature looked as if it was the direct average from maximum and minimum temperatures. This was noticed because whenever a value was missing from any of max or

min temperature values, the average value was also missing. To reduce max, min and average into a single feature, we interpolated all three and took the median. (This worked better in models rather than using min or max alone.)

As noticed in the correlation analysis, 'reanalysis_avg_temp_k', 'reanalysis_sat_precip_amt_mm', 'reanalysis_specific_humidity_g_per_kg' were removed.

Model selection

MANUAL MODEL SELECTION

First, Neural Network based models were used since we did not understand the dataset in the beginning of the project. Since Neural Networks are good at automatically identifying correlations among data, and its immunity to noise in data, it was selected as the first approach to solving the problem.

Many models were tested against the dataset using simple model evaluation tools. From the test results yielded by the tools, it was evident that decision tree methods-based models tend to perform better compared to others. Among the decision tree-based models, random models outperformed the models without randomness. As the result, Random Tree and Random Forest algorithms were selected to be used for the predictions. Since Random Forest is an ensemble of multiple decision trees, it outperformed the Random Tree algorithm.

AUTOMATED MODEL SELECTION

A cross-validated grid search script that was based on sklearn grid search was used to determine best model using resultant 5-fold cross validated MAEs. This script also selects best hyperparameters automatically. Often, for most features, AdaBoost regressor was selected as the best algorithm.

When using Matlab’s regression fitter, Boosted Trees performed better. Although Gaussian Process Regression performed better that Boosted Trees in some cases, when submitted, it resulted in higher MAE than expected. This is probably because the model overfitted the data.

Using Orange3’s (A GUI tool for creating machine learning workflows) test and prediction tool, AdaBoost was selected with least MAE.

BEST MODEL THAT WORKED

Although Boosted Trees, Random Forests and other sophisticated algorithms performed fine, none of them could reach an overall MAE below 24. A dual linear regressor model could reach better scores without any lagging with just using the most correlated features found in the correlation analysis.

Training and evaluation

DEEP NEURAL NETWORKS BASED MODEL

As a first approach to solving the problem, we have used a neural network with four hidden layers, since we needed the neural network to figure out the correlation between data and give a better prediction accuracy. The following diagram shows the neural network model we have trained to solve the problem.

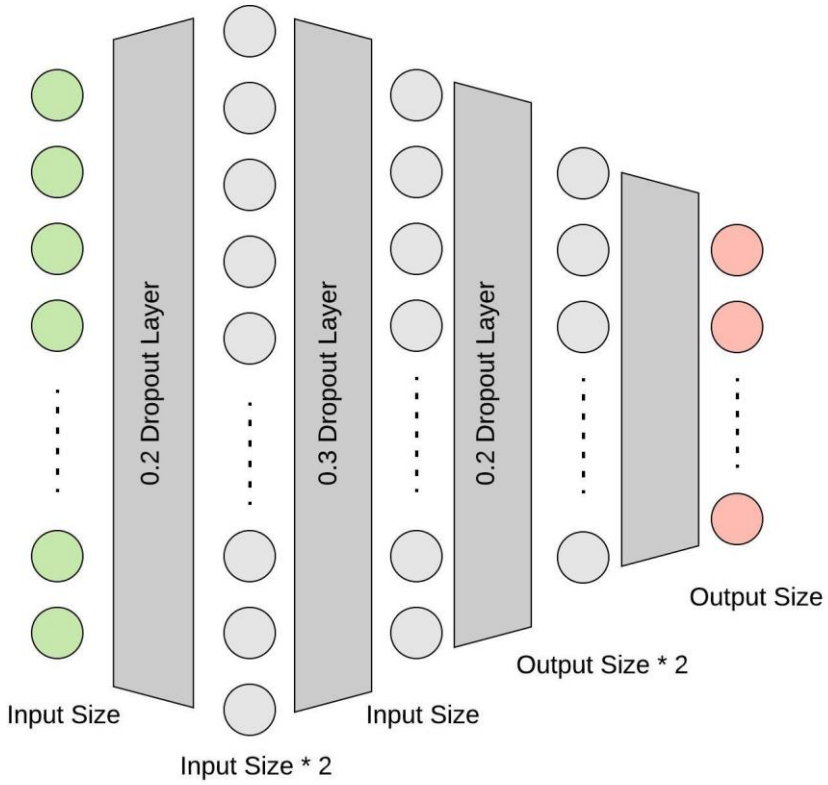


Figure 17 – Neural Network model

Without any interpolation of missing data, the model was able to predict results with MAE approximately 29. Although neural networks can handle noise well, we improved data fed to the neural network model using following methods.

- Input Smoothing using a sliding window of 5

- Output Smoothing using a sliding window of 5
- Separating the data set into 2 cities
- Feature selection (most correlating five features)

Finally, after improving the data set, the model was able to predict results with MAE approximately 25.

RANDOM FOREST

To experiment more using several models, we used the simple model evaluation tool *weka* [3] to experiment with data. The best results were achieved by using decision-tree-based ensemble methods such as Random Forest Regression algorithm. We did the following improvements to the data before feeding the data into the learning model.

- Input Smoothing using a sliding window of 5
- Output Smoothing using a sliding window of 5
- Separating the data set into 2 cities
- Feature selection (most correlating five features)

The model was able to predict results with MAE approximately 24. The model results were comparatively better than our previous attempts.

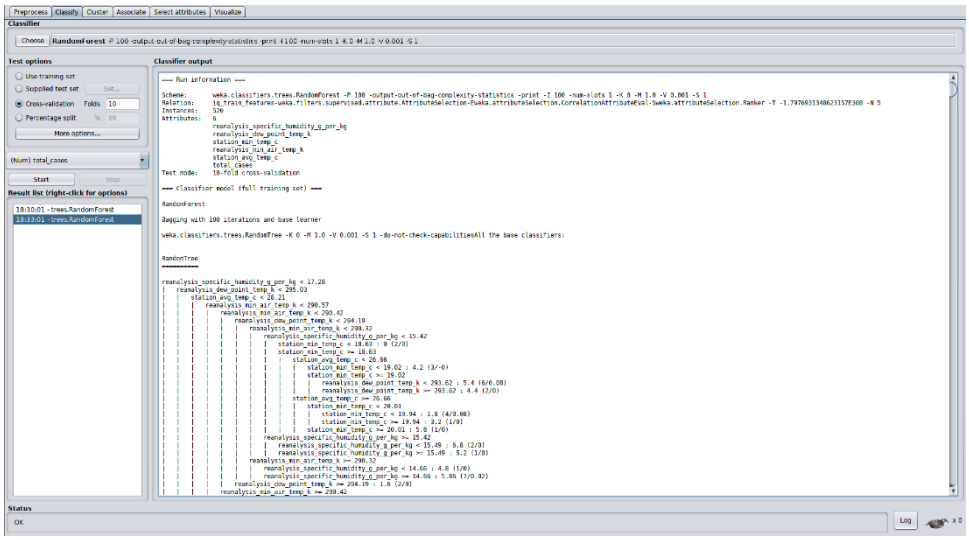


Figure 18 – Model evaluation experimentation with *weka* tool

BOOSTED TREES/ADABOOST

AdaBoost algorithms performed very well predicting the results. The first model that was able to reach a MAE below 25 was a Boosted Trees Regressor. The model used default (cleaned) set of features plus the same features lagged by one week. Smoothing inputs and labels reduced the errors slightly.

BOOSTED TREES WITH STL

Trend and seasonal components of the best correlated features was used and lagged by the amounts found in correlation analysis. Removal of noise and outliers is automatically performed because of this component separation. Although this is a sound idea, the model was not able to reach a score that is at least better than the previous models.

DUAL-LINEAR REGRESSION

First, week of year was used to train a linear regressor to predict the seasonal components of the total cases label. This works because the week numbers are recurring per each year. After trained, the model will be able to predict a general possible value for each week.

When we subtract this predicted value from the original total cases feature, what remains is a trend and noise component that can be predicted using another model.

To predict this trend, the features 'reanalysis_precip_amt_kg_per_m2', 'reanalysis_relative_humidity_percent', 'station_temp' (average of station temperature values) were used. (These features showed the best results when submitted.) These featured are smoothed using mean with a window of 53 or 52 weeks (Some years have 53 weeks while others only have 52. However, 53 worked better). The resulting curve is essentially none other than the trend (long term progression) of the feature. These were then used by another linear regression model to predict the trend component of the total cases, by first training them with the remainder of total cases resulted by the previous calculation. The predicted trend and seasonal components are then added to get the result.

Parameter tuning

A cross validated grid search was used to find the correct hyperparameters of the models. This grid search trains the model using randomly chosen 80% of the data and uses 20% remaining data (since we used 5-fold cross validation) for validation. This is done separately until all the data is used for training and validation. We can give the ranges for the model hyperparameters such as alpha, learning rate etc. All the combinations of the hyperparameters are tested and the values that result in the best model are returned.

Prediction

Following is a summary of predictions done using different algorithms.

Model	Description	Typical results when submitted (MAE)
DNN	Deep (with multiple hidden layers) neural networks	24 to 29
Random Forest	Random Forest Regressors with few relevant features	24+
AdaBoost/Boosted Trees	Different software/algorithms of boosting with all features, one week lagged, or most relevant features	24+
Boosted Trees with STL	Automatically selected most relevant STL-decomposed seasonal and trend components lagged to maximize correlation with total cases	25 to 35+
Linear regression (dual)	Dual-linear regression models composed to predict seasonal and trend components of total cases	Below 22, 19.3798 (best result so far)

DISCUSSION

We submitted a total of 27 results to the dengAI competition.most of them are being models that use regression trees.

Even though a week is a long time, noise was present in the dataset. This was clear since smoothing significantly increased the prediction accuracy. For example, in the first AdaBoost models, smoothing (5 to 7 weeks window/mean) resulted in better accuracy than lagging the data alone by 1 week.

Sometimes, when highly cross-correlated features are removed, boosted tree models would yield a worse result. This is an indication that the model did not exactly fit the purpose.

First, a Neural networks model was used to do the predictions because we did not know the correlations among the data well, and since neural networks are good at identifying hidden correlations among data. The model handled noise well, and the predictions were fairly

accurate. But, in order to train a good neural network model, the given data set size was not large enough. When the amount of data was not enough, the model tends to overfit for the given test data and had a very lower performance when predicting results for unseen data. So, we came into a conclusion that the problem needs to be solved using efficient ways which can predict better results with less amount of data.

The second set of models that had been tested were the Boosted Tree Regressors. They easily outperformed neural networks. They were fairly resistant to noice too although smoothing gave better results. AdaBoost and other boosted trees often were the selected model by automatic model selection.

When seasonal decomposition was used to preprocess the feature set before training boosted trees often resulted in worse results although the patterns in the data were clearly visible even to the naked eye. This can either be because of the wrong model selection or because we selected models using validation results but found the correlation and seasonal decomposition using only statistical methodologies.

The Random Forest algorithm was used as the another attempt to predict the results. This was choosen because the algorithm uses multiple models to obtain better predictive performance by using ensembling methods. Although the results were better than neural networks, it was not good enough compared to the scores of the other competitors. So, it was required to improve the models and carefully select the data in order to achieve better prediction accuracy.

The model that worked best was non-other than simple linear regression. But the way it was used is different as mentioned earlier. It is noticeable that the reason behind why the traditional seasonal decomposition not worked but the seasonal decomposition using a linear regressive learned model should be some inconsistency between traditional statistical decomposition vs. learned models. However, even for this 2-model approach, complex algorithms like Random Forest did not give better results than linear regression.

CONCLUSION

In our project we created predictive models of dengue epidemic in connection with the US Department of Commerce Dengue Forecasting project and the “DengAI: Predicting Disease Spread” competition from DrivenData.

We used neural network models to predict dengue cases and compared them with few regression methods. We submitted our results to the competition to find out that neural network based methods performed poorly relative to few other traditional ML models we tried.

However, it should be noted that most models predicted the number of cases, even with minimal human intervention, with a good enough accuracy for all practical purposes such as preparing for an epidemic. The importance of data mining and machine learning is emphasized by competitions like this.

REFERENCES

[1] "DengAI: Predicting Disease Spread", DrivenData, 2018. [Online]. Available: https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/ . [Accessed: 20- Jul- 2018].

[2] “statsmodels”, Statsmodels, 2018. [Online]. Available https://github.com/statsmodels/statsmodels/blob/v0.9.0/statsmodels/tsa/seasonal.py#L89.

[3] "Weka 3 - Data Mining with Open Source Machine Learning Software in Java", *Cs.waikato.ac.nz*, 2018. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka. [Accessed: 20- Jul- 2018].

***