*Natalie Carlson*

*CSCE 490 | Final Project Report*

*Dengue Fever*

**Project description**

Using environmental data collected by various U.S. Federal Government agencies I sought to predict the number of dengue fever cases reported each week in San Juan, Puerto Rico and Iquitos, Peru. My predictions are based on environmental variables describing changes in temperature, precipitation, vegetation, and more.

**Background**

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.

Because it is carried by mosquitoes, the transmission dynamics of dengue are related to climate variables such as temperature and precipitation. Although the relationship to climate is complex, a growing number of scientists argue that climate change is likely to produce distributional shifts that will have significant public health implications worldwide.

In recent years dengue fever has been spreading. Historically, the disease has been most prevalent in Southeast Asia and the Pacific islands. These days many of the nearly half billion cases per year are occurring in Latin America.

Accurate dengue predictions would help public health workers and people around the world take steps to reduce the impact of these epidemics. But predicting dengue is a hefty task that calls for the consolidation of different data sets on disease incidence, weather, and the environment. My goal was to predict the total\_cases label for each (city, year, weekofyear) in the data test set.

**Data**

The data for this project was provided by DrivenData as part of a machine learning competition and can be accessed at <https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/page/80/>. The data comes from multiple sources aimed at supporting the Predict the Next Pandemic Initiative. Dengue surveillance data is provided by the U.S. Centers for Disease Control and prevention, as well as the Department of Defense's Naval Medical Research Unit 6 and the Armed Forces Health Surveillance Center, in collaboration with the Peruvian government and U.S. universities. Environmental and climate data is provided by the National Oceanic and Atmospheric Administration (NOAA), an agency of the U.S. Department of Commerce.

There are two cities, San Juan and Iquitos, with data for each city spanning between 1990 and 2010. The features and output are on a (year, weekofyear) timescale and missing values were filled as Nans.

**Features**

### City and date indicators

* 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](https://www.ncdc.noaa.gov/oa/climate/ghcn-daily/) weather station measurements

* 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](http://www.ncdc.noaa.gov/cdr/operationalcdrs.html) (0.25x0.25 degree scale)

* precipitation\_amt\_mm – Total precipitation

### NOAA's NCEP [Climate Forecast System Reanalysis](http://rda.ucar.edu/datasets/ds093.0/#metadata/detailed.html?_do=y) measurements (0.5x0.5 degree scale)

* 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](https://www.ncdc.noaa.gov/cdr) (0.5x0.5 degree scale) measurements

* 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

**Pre-processing**

The data for features and output were in separate csv files, so the first thing I did was merge them into a single data frame. Review of data types showed ‘city’ and ‘week\_start\_date’ were both objects so I converted ‘city’ to a category and then to an int code, then because ‘week\_start\_date’ was redundant to ‘year’ and ‘weekofyear’ I chose to drop the feature from the data frame. I next dropped any records that were completely empty or had no output value. Finally, I filled all Nan valued cells with data from the cell above (i.e. the prior week) in an attempt to get as close to the true value as possible.

To determine features with extreme outliers, I applied a z-score with a threshold of 5 with the following results: 'station\_precip\_mm' 15 outliers, 'station\_precip\_mm' 7 outliers, ‘station\_min\_temp\_c’ 5 outliers, ‘reanalysis\_min\_air\_temp\_k’ 3 outliers, and 'reanalysis\_relative\_humidity\_percent' 3 outliers. I then used boxplots to visualize the outliers and determine a maximum/minimum data cutoff for each of the five features which resulted in the elimination of twenty records in the data set.

|  |  |  |
| --- | --- | --- |
|  | A close up of a logo  Description automatically generated | A close up of a logo  Description automatically generated |
| A screenshot of a cell phone  Description automatically generated | A close up of a logo  Description automatically generated | A close up of a logo  Description automatically generated |

When performing machine learning with regression models, one or two variables may have much higher values than the others and can dominate thereby reducing accuracy. A statistical description of the data along with a visualization (see below) indicated a need to normalize the data. This was done using sklearn’s StandardScaler that was applied to each model.

A screen shot of an open window

Description automatically generated

My final steps in pre-processing my data was to split it into training and test sets using sklearn’s train\_test\_spllit. I chose a test size of 20% and random state 42.

**Models**

Because my output was continuous, I used regression models. Initially, I fit linear regression, decision tree, random forest, multilayer perceptron, and k-nearest neighbor regression. I then calculated the Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and R-Squared Score (R2) on each model to assess the most appropriate model for the data set with the following results:

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

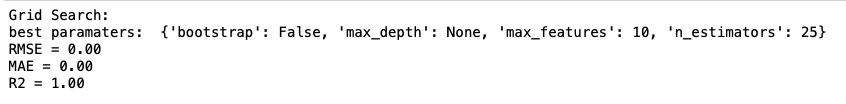
Description automatically generated

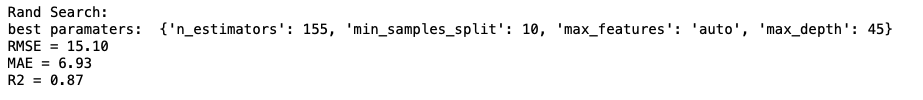
The decision tree model had the best results, however, because decision trees are prone to overfitting, I selected random forest for my model.

**Training and Testing**

To fine tune the random forest model, I conducted both a grid search and random search using sklearn’s model\_selection library. For the grid search I set the following parameters: number of trees: n\_estimators=[10, 25], max number of features for splitting a node: max\_features=[5, 10], max number of levels in each tree: max\_depth= [10, 50, None], and replacement: bootstrap=[True, False]. For the random search I set the parameters as follows: number of trees: n\_estimators=(between 20 and 200), min number of data points placed in a node before the node is split: min\_samples\_split= (between 5 and 10), max number of features for splitting a node: max\_features=(auto or sqrt), and max number of levels in each tree: max\_depth=(between 1 and 45).

For each search type, I ran a Mean Test Score(MTS) and calculated the best estimator. Next, I computed the RMSE, MAE, and R2 on the parameters of each search type’s best estimator.





Having found the best parameters, I fit the test set to three models- random forest, random forest with the best grid search parameters, and random forest with the best random search parameters. The model with grid search received perfect scores for RMSE, MAE, and R2, however, because grid search computation time is costly, I wanted to see if I could fine tune the model even further by reducing the number of features. Using matplotlib I produced a visualization of the correlation of features and I created a table showing the feature importance of the grid search and random search models using sklearn’s feature\_importances\_ library.A picture containing drawing

Description automatically generated

A close up of a piece of paper

Description automatically generated

After creating a new dataframe with just the top four features ('year', 'weekofyear','ndvi\_se', 'ndvi\_sw'), I re-ran the metrics for the standard random forest and saw improvement. RMSE decreased to 6.96 from 8.26, MAE decrease to 3.96 from 4.46, and the R2 increased to 0.97 from 0.96.

**Conclusion**

Using a random forest regression model with environmental data collected by various U.S. Federal Government agencies I was able to predict the number of dengue fever cases reported each week in San Juan, Puerto Rico and Iquitos, Peru with near perfect accuracy. Of the 22 features provided, only four (the year, week of year, and satellite vegetation data in the southern portions of each city) are needed to get an R2 score of 97% on the standard random forest regression model, and a random forest with grid search can achieve 100%.