# PROJECT REPORT

## CS 6375.501 – DengAI Predicting Disease Spread Machine Learning – Project Report

*Kassap, Christopher (cxk112830)*

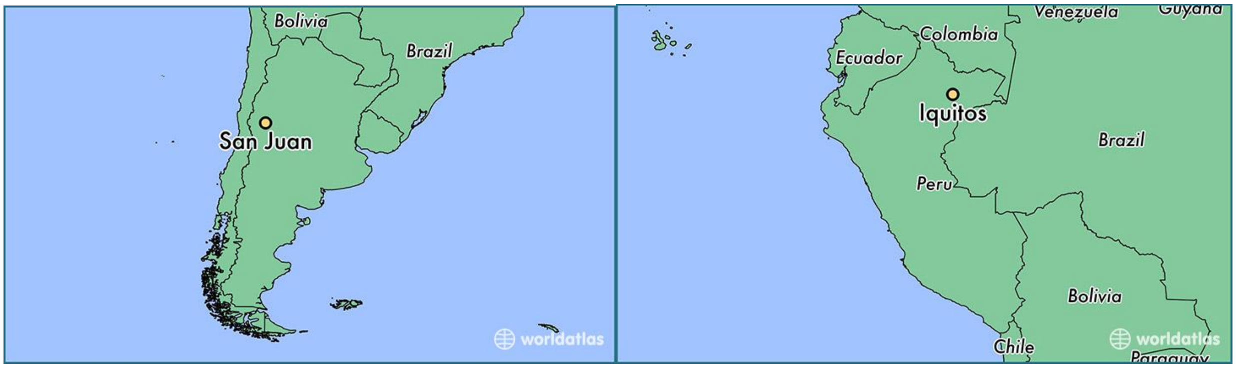
*Vutukuru, Ram Anand (rxv162130)*

*Kataria, Jaiminee (jxk172330)*

*Kaneria, Dhwaniben Rameshbhai (drk170130)*

# Introduction:

DengAI is a machine learning project with the goal of predicting the spread of Dengue fever across the globe. Specifically, the aim of the project is to predict the next pandemic of the disease before it occurs in San Juan, Puerto Rico or Iquitos, Peru. Dengue fever is primarily transmitted through mosquitos carrying the disease, and it is therefore highly dependent on climate and vegetation factors.




The learner takes input environmental and climate data provided by the National Oceanic and Atmospheric Administration (NOAA) and Centers for Disease Control and Prevention, and outputs the total number of dengue fever cases reported each week for a given year in either San Juan or Iquitos.

# Dataset Description:

**Name**: DengAI database
**Number of Instances**: 1456
**Class Components**:

- city : city abbreviations. Two types: sj for San Juan, and iq for Iquitos
- year : the year. yyyy format
- weekofyear : the week of the year. mm/dd/yyyy format.
- week_start_date : the week of year's start date. Mm/dd/yyyy format.
- total_cases : the total number of cases. Integer.

**Number of Attributes**: 20
**Attribute Types**: Real
**Attributes**:

NOAA's CDR Normalized Difference Vegetation Index (NVDI) measurements

- ndvi_ne : pixel southeast of city centroid
- ndvi_nw : pixel southwest of city centroid
- ndvi_se : pixel northeast of city centroid
- ndvi_sw : pixel northwest of city centroid

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

- precipitation_amt_mm : total precipitation

NOAA's NCEP climate forecast system reanalysis

- reanalysis_air_temp_k : mean air temperature
- reanalysis_avg_temp_k : average 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 temprature
- reanalysis_precip_amt_kg_per_m2 : total precipitation
- reanalysis_relative_humidity_percent : mean relative humidity
- reanalysis_sat_precip_amt_mm : total precipitation
- reanalysis_specific_humidity_g_per_kg : mean specific humidity
- reanalysis_tdtr_k : diurnal temperature range

NOAA's GHCN daily climate data weather station measurements

- station_avg_temp_c : average temperature
- station_diur_temp_rng_c : diurnal temperature range
- station_max_temp_c : maximum temperature
- station_min_temp_c : minimum temperature
- station_precip_mm : total precipitation

# Preprocessing:

**Missing Values**:

We have removed NaN values and filled them with previous value in every column feature.

```
ProcessedData = result.apply(lambda x: x.fillna(method='ffill'))
```

NanRemoval.xlsx

**Conversion from Kelvin to Celsius**:

All temperatures were converted to Celsius in order to maintain consistency among different measures:

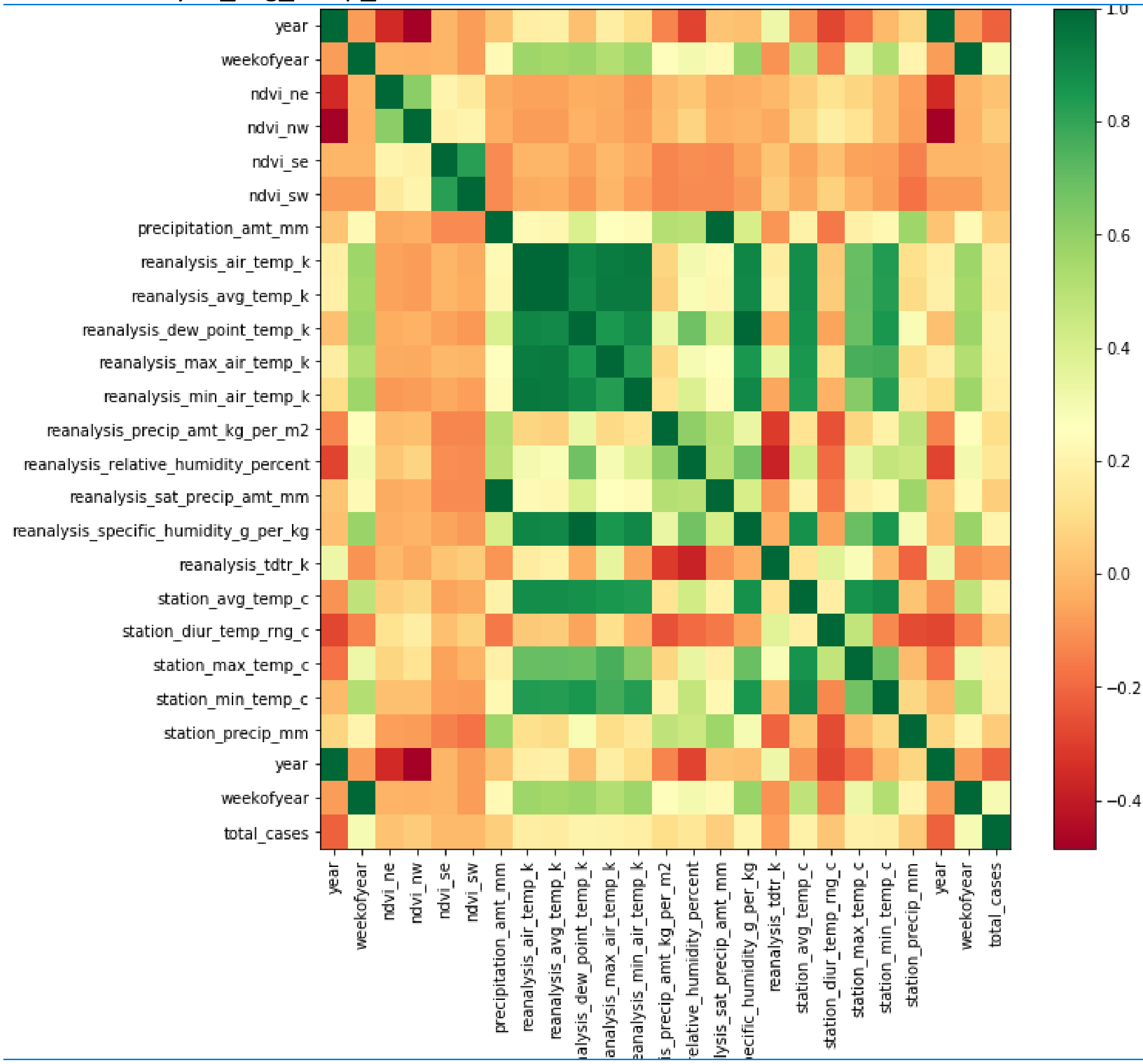
```
c=["reanalysis_air_temp_k", "reanalysis_avg_temp_k","reanalysis_dew_point_temp_k",
"reanalysis_max_air_temp_k","reanalysis_min_air_temp_k"]
for i in c:
  ProcessedData[i] = ProcessedData[i] - 273.15
```

**Removal of Redundant Attributes Based on Correlation**:

Here we can see that none of the variables are correlated with total cases, but some are correlated with each other. Therefore we have deleted those features based on the following plot.

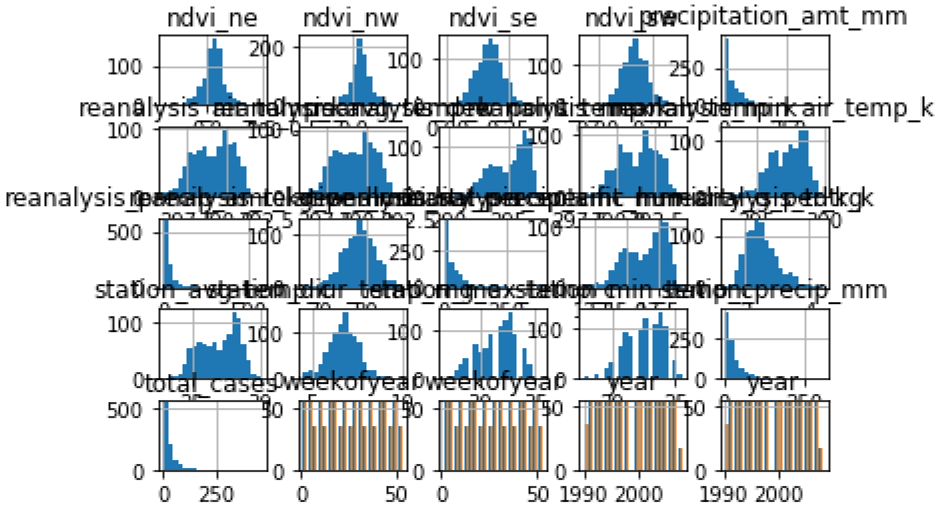
reanalysis_specific_humidity_g_per_kg
reanalysis_dew_point_temp_k
reanalysis_sat_precip_amt_m
precipitation_amt_mm

reanalysis_air_temp_k
reanalysis_avg_temp_k




**Removal of High Variance Features**:

The features with the highest variance will affect the prediction of total cases, and therefore tarnish the results. To determine which features had the highest variance, we plotted histogram and various graphs to observe which features spiked. After determining the features with high variance, we removed them from the data.




```
dtype: float64
ndvi_ne --> 0.019189678045667725
ndvi_nw --> 0.014294789737666872
ndvi_se --> 0.005536768213180817
ndvi_sw --> 0.006987651634564899
precipitation_amt_mm --> 1916.6287044248636
reanalysis_air_temp_k --> 1.8549088683244301
reanalysis_avg_temp_k --> 1.5943513892124044
reanalysis_dew_point_temp_k --> 2.333338523633496
reanalysis_max_air_temp_k --> 10.452485253578029
reanalysis_min_air_temp_k --> 6.5505184047052785
reanalysis_precip_amt_kg_per_m2 --> 1877.4173028932078
reanalysis_relative_humidity_percent --> 51.2801537887176
reanalysis_sat_precip_amt_mm --> 1916.6287044248636
reanalysis_specific_humidity_g_per_kg --> 2.378616121931116
reanalysis_tdtr_k --> 12.54816917163751
station_avg_temp_c --> 1.6406037993486386
station_diur_temp_rng_c --> 4.5064341978139515
station_max_temp_c --> 3.8446153798950236
station_min_temp_c --> 2.4621168196065084
station_precip_mm --> 2243.2826944176977
city --> Series([], dtype: float64)
year --> year    29.24986
year    29.24986
dtype: float64
weekofyear --> weekofyear    225.583493
weekofyear    225.583493
```

**Feature Engineering**:

It is the process to make Algorithms of Machine Learning efficient by introducing new features to the dataset or transforming our features.

From observing the dataset, we found that cases recorded for a given week are not the result of that week but of the previous week. The most likely reason for this is that the incubation period is 4-7 days. Therefore the infection in the given week is directly related to previous week.

To apply this on the dataset, we shifted data by one week. We experimented with shifting data by 1 week, 2 weeks, and 3 weeks to get an idea regarding how the data pattern is effected by time period.

| reanalysis_tdtr_k | station_avg_temp_c | station_diur_temp_rng_c | station_max_temp_c | station_min_temp_c | station_precip_mm | total_cases | Lag_by_1_Week | Lag_by_2_Weeks |
|---|---|---|---|---|---|---|---|---|
| 2.628571 | 25.442857 | 6.900000 | 29.4 | 20.0 | 16.0 | 4 | 5 | 4 |
| 2.371429 | 26.714286 | 6.371429 | 31.7 | 22.2 | 8.6 | 5 | 4 | 3 |
| 2.300000 | 26.714286 | 6.485714 | 32.2 | 22.8 | 41.4 | 4 | 3 | 6 |
| 2.428571 | 27.471429 | 6.771429 | 33.3 | 23.3 | 4.0 | 3 | 6 | 2 |
| 3.014286 | 28.942857 | 9.371429 | 35.0 | 23.9 | 5.8 | 6 | 2 | 4 |
| 2.100000 | 28.114286 | 6.942857 | 34.4 | 23.9 | 39.1 | 2 | 4 | 5 |
| 2.042857 | 27.414286 | 6.771429 | 32.2 | 23.3 | 29.7 | 4 | 5 | 10 |
| 1.571429 | 28.371429 | 7.685714 | 33.9 | 22.8 | 21.1 | 5 | 10 | 6 |
| 1.885714 | 28.328571 | 7.385714 | 33.9 | 22.8 | 21.1 | 10 | 6 | 8 |
| 2.014286 | 28.328571 | 6.514286 | 33.9 | 24.4 | 1.1 | 6 | 8 | 2 |
| 2.157143 | 27.557143 | 7.157143 | 31.7 | 21.7 | 63.7 | 8 | 2 | 6 |

# Proposed Solution:

**Model Training and Validation:**

We are using below Performance metrics.

Mean Absolute Error:

MAE measures the average magnitude of the errors in a set of predictions, without considering their direction.

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

**Root Mean Square Error**:

RMSE is a quadratic scoring rule that also measures the average magnitude of the error.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

# Experimental Results:

**Classifiers**:

**Neural Network (Lag by 1 week):**

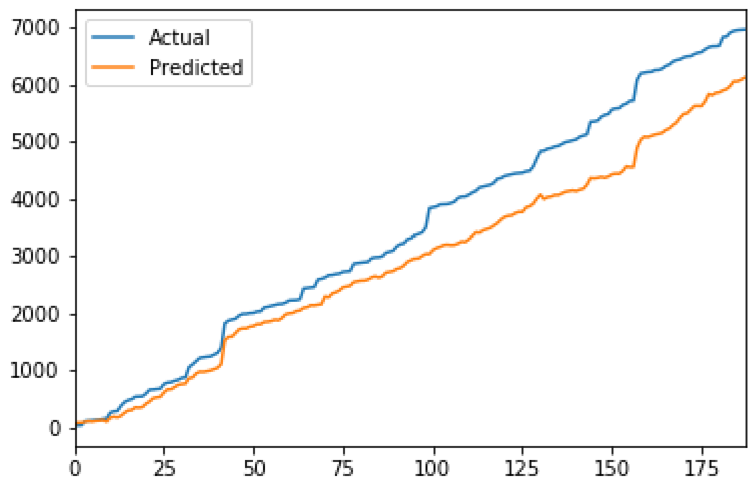
Trials:

| Classifier | Parameters | MAE | RMSE |
|---|---|---|---|
| Neural Network | mlp = MLPRegressor(solver='lbfgs', hidden_layer_sizes=50, max_iter=200,random_state=1)<br><br>#'lbfgs' is an optimizer in the family of quasi-Newton method | IQ: 9.75997853981<br>SJ:27.068202278731 | IQ: 13.92249974226695<br>SJ:43.10857507232615497 |
| Neural Network | mlp = MLPRegressor(solver='sgd', hidden_layer_sizes=50, max_iter=200,random_state=1)<br><br>#'sgd' refers to stochastic gradient descent | IQ:7.27837992821267127<br>SJ: 23.277988834165 | IQ:15.38784521322963417<br>SJ:37.38982155913268046 |
| Neural Network | mlp = MLPRegressor(solver='lbfgs', hidden_layer_sizes=100, max_iter=200,random_state=1)<br><br>#hidden layers = 100 | IQ:10.108910380901<br>SJ: 27.111943343208 | IQ:15.15395349818083812<br>SJ:41.17011204148125025 |
| Neural Network | mlp = MLPRegressor(solver='lbfgs', hidden_layer_sizes=100, max_iter=200,random_state=1,activation='logistic')<br><br># activation ='logistic', the logistic sigmoid function, returns f(x) = 1 / (1 + exp(-x)) | IQ:9.212904378769<br>SJ: 39.631886079487 | IQ:12.6496006068542544<br>SJ: 53.5583495404022034 |
| Neural Network | mlp = MLPRegressor(solver='lbfgs', hidden_layer_sizes=50, max_iter=100,random_state=1) | IQ:10.128493350741<br>SJ: 25.172760708502 | IQ:14.17595035469161588<br>SJ: 41.2645672997759061 |

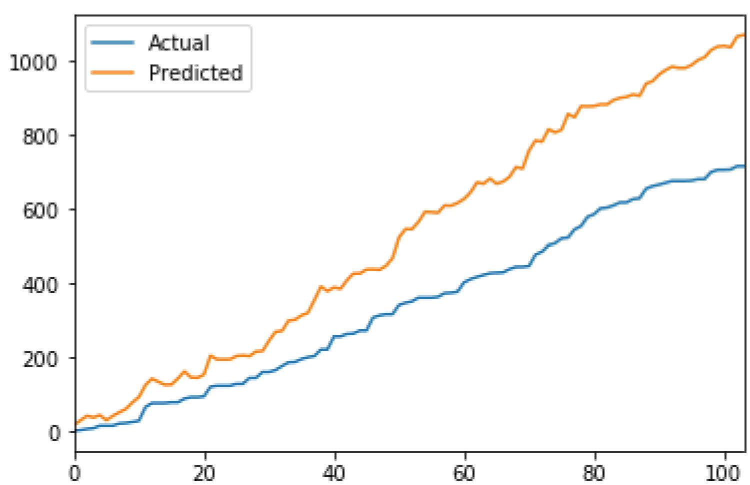
| | | | |
|---|---|---|---|
| | #max iteration=100 | | |

Predicted vs Actual plots:

San Ivan




Iquitos




**Neural Network (Lag by 2 weeks):**

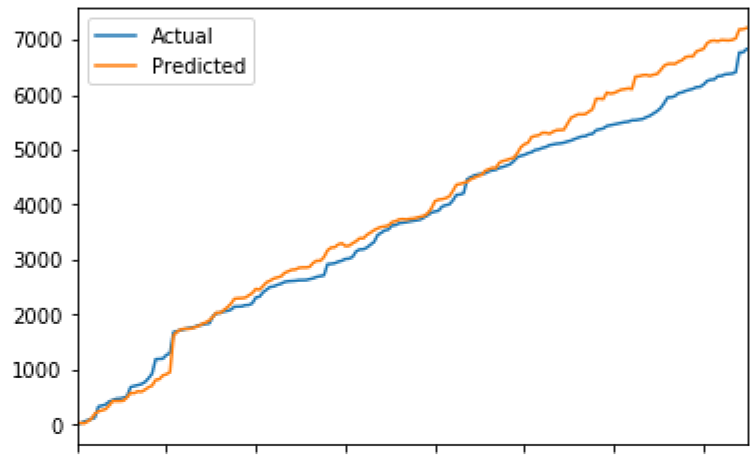
Trials:

| Classifier | Parameters | MAE | RMSE |
|---|---|---|---|

| | | | |
|---|---|---|---|
| Neural Network | mlp = MLPRegressor(solver='lbfgs', hidden_layer_sizes=50, max_iter=200,random_state=1)<br><br>#'lbfgs' is an optimizer in the family of quasi-Newton method | IQ:10.9747584293<br>SJ:27.3830359996 | IQ: 14.62640424443986<br>SJ: 43.47100427667421 |
| Neural Network | mlp = MLPRegressor(solver='sgd', hidden_layer_sizes=50, max_iter=200,random_state=1)<br><br>#'sgd' refers to stochastic gradient descent | IQ:7.38500030016<br>SJ:21.1046204624 | IQ:12.504420563797925<br>SJ: 32.33389546553753 |
| Neural Network | mlp = MLPRegressor(solver='lbfgs', hidden_layer_sizes=100, max_iter=200,random_state=1)<br><br>#hidden layers = 100 | IQ: 10.570712085<br>SJ:32.6859740247 | IQ: 14.18484769874857<br>SJ: 57.14716823623551 |
| Neural Network | mlp = MLPRegressor(solver='lbfgs', hidden_layer_sizes=100, max_iter=200,random_state=1,activation='logistic')<br><br># activation ='logistic', the logistic sigmoid function, returns f(x) = 1 / (1 + exp(-x)) | IQ:13.1256304839<br>SJ:30.5763356839 | IQ: 18.84256435788437<br>SJ: 43.24548565132035 |
| Neural Network | mlp = MLPRegressor(solver='lbfgs', hidden_layer_sizes=50, max_iter=100,random_state=1)<br><br>#max iteration=100 | IQ:9.2529503918<br>SJ:30.7310484535 | IQ:14.463560069904457<br>SJ: 50.78768510108033 |

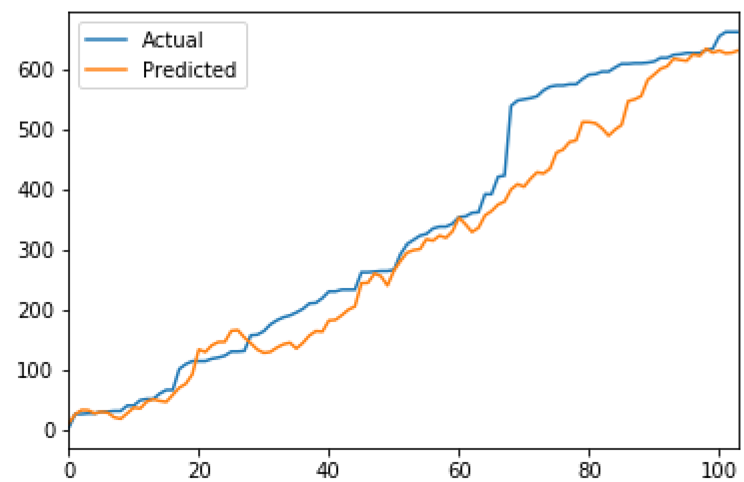
Predicted vs Actual plots:

San Ivan




matplotlib.figure.Figure at 0x2e8fbbde1d0>

Iquitos




K- NN (**Lag By 1 Week):**

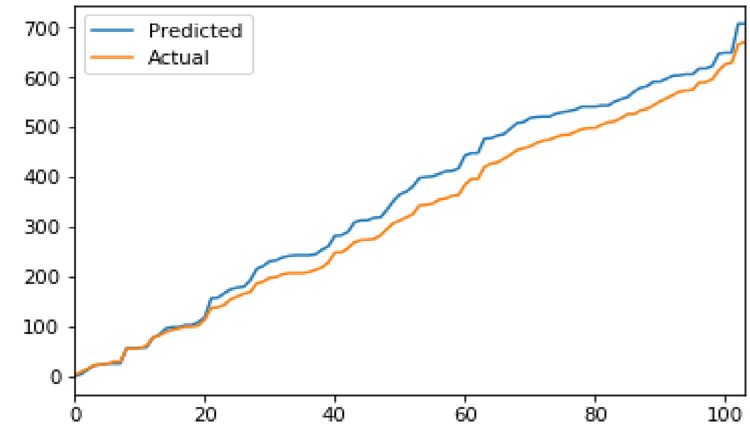
Trials:

| Classifie r | Model | MAE | RMSE |
|---|---|---|---|
| K-NN | neighbors.KNeighborsRegressor(5, weights='uniform') | IQ=7.888<br>SJ=18.52553 | IQ=13.71091<br>SJ=35.94289 |
| K-NN | neighbors.KNeighborsRegressor(10, weights='uniform',leaf_size=10) | IQ=7.47211<br>SJ=18.35106 | IQ=12.28658<br>SJ=36.3672 |
| K-NN | neighbors.KNeighborsRegressor(5, weights='distance',leaf_size=10) | IQ=7.204552<br>SJ=18.36947 | IQ=13.64428<br>SJ=33.79175 |
| K-NN | neighbors.KNeighborsRegressor(3, weights='uniform',algorithm='kd_tree') | IQ=8.11858<br>SJ= 18.9609 | IQ=15.64428<br>SJ=36.28426 |

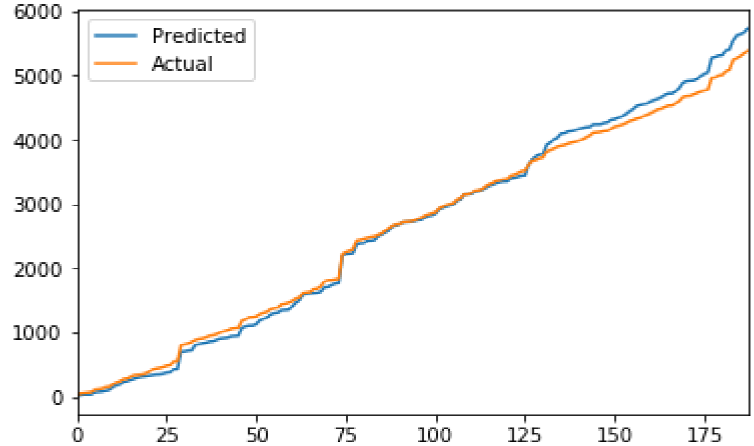
| | | | |
|---|---|---|---|
| K-NN | neighbors.KNeighborsRegressor(10, weights='distance',algorithm='ball_tree',leaf_size=10) | IQ=7.40422 SJ=19.1083 | IQ=12.16155 SJ=34.96233 |

Predicted vs Actual plots:

San Juan




Iquitos




**Neural Network (Lag by 2 Weeks):**

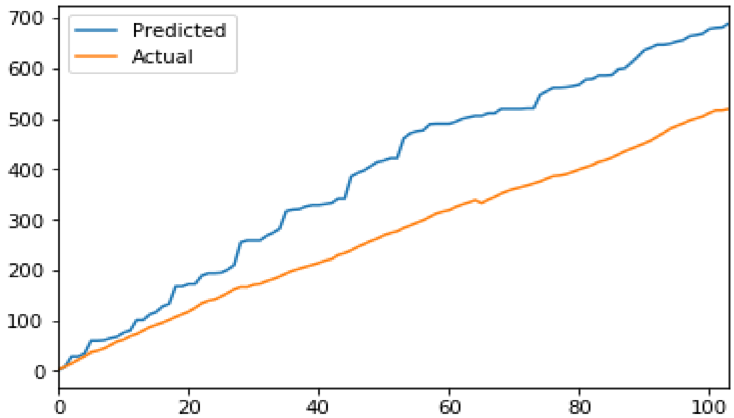
Trials:

| Classifier | Model | MAE | RMSE |
|---|---|---|---|
| K-NN | neighbors.KNeighborsRegressor(5, weights='uniform') | IQ=6.311538 SJ=20.81170 | IQ= 9.6197 SJ=37.82495 |
| K-NN | neighbors.KNeighborsRegressor(10, weights='uniform',leaf_size=10) | IQ= 6.03461 SJ=23.76569 | IQ= 9.04612 SJ=41.87223 |
| K-NN | neighbors.KNeighborsRegressor(5, weights='distance',leaf_size=10) | IQ= 6.28579 SJ=20.74901 | IQ= 9.52173 SJ=37.42315 |
| K-NN | neighbors.KNeighborsRegressor(3, weights='uniform',algorithm='kd_tree') | IQ= 6.85256 SJ= 21.5607 | IQ=4.64428 SJ=38.55126 |
| K-NN | neighbors.KNeighborsRegressor(10, | IQ= 6.02845 SJ=33.46444 | IQ= 9.01131 SJ=63.42415 |

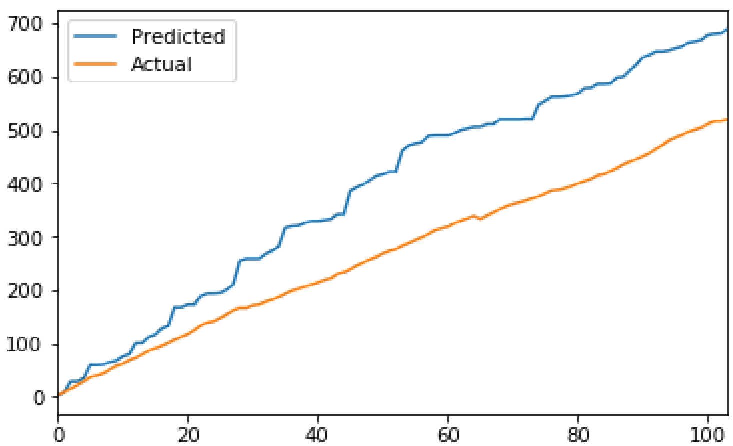
| | | | |
|---|---|---|---|
| | weights='distance',algorithm='ball_tree',leaf_size=10) | | |

Predicted vs Actual plots:

Iquitos




San Juan




**SVM (Lag by 1 Week):**

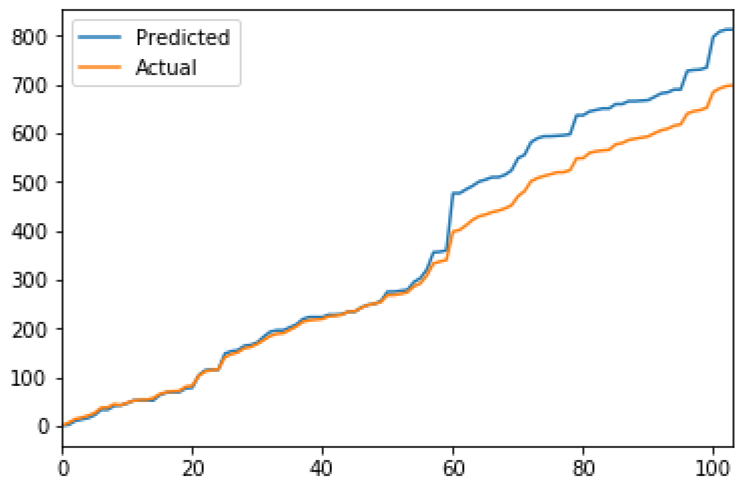
Trials:

| Classifier | Model | MAE | RMSE |
|---|---|---|---|
| SVM | SVR(C=1.0, epsilon=0.2) | IQ=6.960261 | IQ=12.66123 |

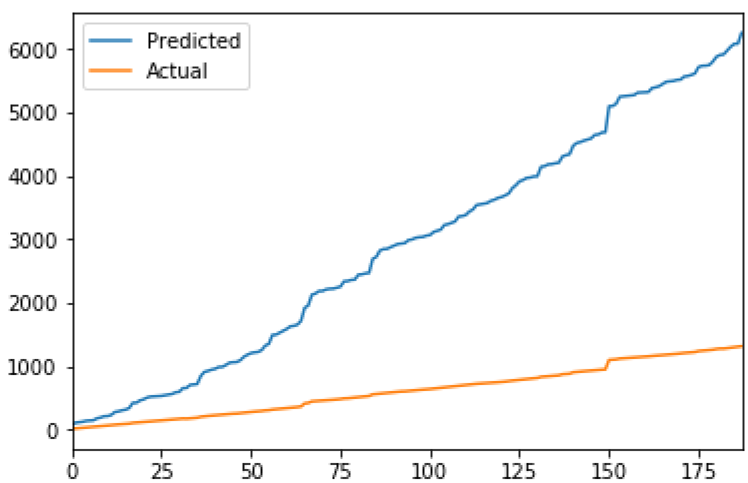
| | | | |
|---|---|---|---|
| | | SJ=27.67210 | SJ=51.27532 |
| SVM | SVR(C=1.5, epsilon=0.1,kernel='poly') | IQ= 13.1334<br>SJ=26.6358 | IQ=15.9210<br>SJ=45.51520 |
| SVM | SVR(C=1.0,kernel='poly',max_iter=10) | IQ= 6.53188<br>SJ=30.01672 | IQ=12.63827<br>SJ= 49.8904 |
| SVM | SVR(C=1.0, verbose=True) | IQ= 6.95372<br>SJ=27.6675 | IQ=12.6612<br>SJ=51.26502 |
| SVM | SVR(C=0.5, kernel='sigmoid') | IQ=6.89750<br>SJ=26.9660 | IQ=12.27729<br>SJ=48.34541 |

Predicted vs Actual plots:

San Juan




Iquitos




**SVM (Lag by 2 Weeks):**

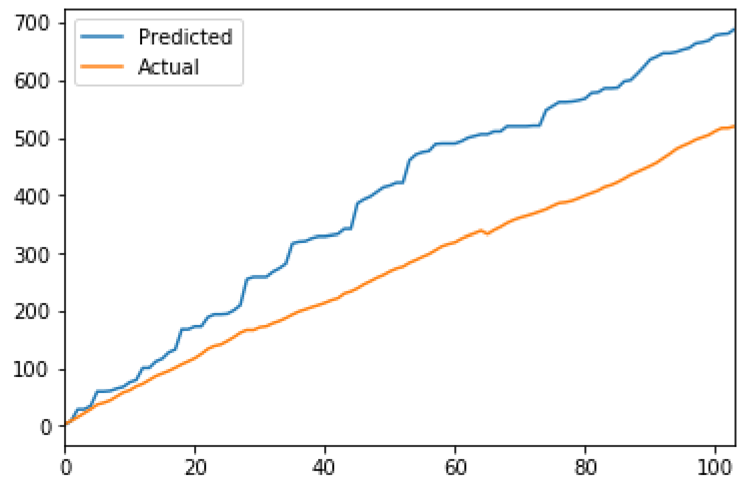
Trials:

| Classifie r | Model | MAE | RMSE |
|---|---|---|---|
| SVM | SVR(C=1.0, epsilon=0.2) | IQ=4.841994<br>SJ=33.4644 | IQ=8.94209<br>SJ=63.42416 |

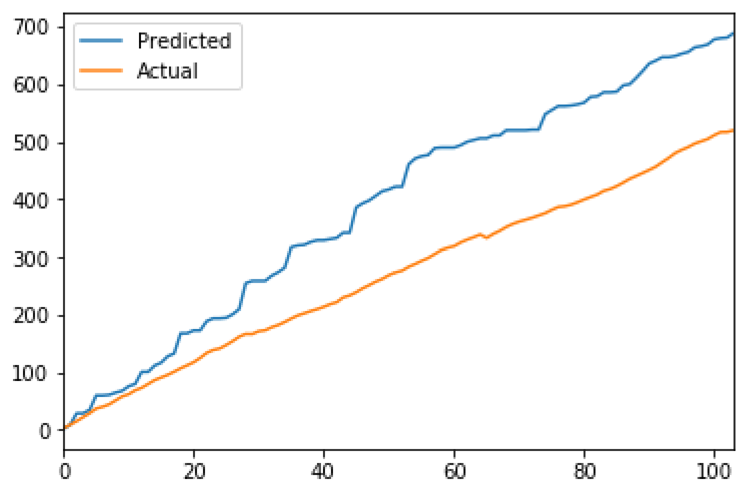
| | | | |
|---|---|---|---|
| SVM | SVR(C=1.5, epsilon=0.1,kernel='poly') | IQ= 5.10118<br>SJ=33.3466 | IQ=8.962202<br>SJ=63.35114 |
| SVM | SVR(C=1.0,kernel='poly',max_iter=10) | IQ=12.63103<br>SJ=27.46658 | IQ=13.42901<br>SJ= 58.2205 |
| SVM | SVR(C=1.0, verbose=True) | IQ=4.843900<br>SJ=33.45845 | IQ=8.94709<br>SJ=63.42054 |
| SVM | SVR(C=0.5, kernel='sigmoid') | IQ=5.087049<br>SJ=33.31485 | IQ=8.96769<br>SJ=63.52483 |

Predicted vs Actual plots:

San Juan




Iquitos




**Gradient Boosting (Lag by 1 week):**

Trials:

| Classifier | Model | MAE | RMSE |
|---|---|---|---|
| Gradient Boosting | params = {'n_estimators': 500, 'max_depth': 4, 'min_samples_split': 2, 'learning_rate': 0.01, 'loss': 'ls'} | IQ:4.8968295205631<br>SJ: 16.52790252061 | IQ: 7.667712237338<br>SJ: 28.85327068511 |

| | | | |
|---|---|---|---|
| Gradient Boosting | params = {'n_estimators': 500, 'max_depth': 5, 'min_samples_split': 3, 'learning_rate': 0.01, 'loss': 'ls'} | IQ:4.5360060217123<br>SJ:14.762399812583 | IQ: 7.4560619784048<br>SJ: 26.054642260933 |
| Gradient Boosting | params = {'n_estimators': 500, 'max_depth': 4, 'min_samples_split': 2, 'learning_rate': 0.01, 'loss': 'lad'} | IQ:4.3231161692191<br>SJ: 15.825258711638 | IQ: 8.01272526993<br>SJ: 36.84445526090 |
| Gradient Boosting | params = {'n_estimators': 500, 'max_depth': 5, 'min_samples_split': 2, 'learning_rate': 0.01, 'loss': 'lad', 'criterion':'friedman_mse'} | IQ:4.2687605543857<br>SJ:15.740664091309 | IQ:7.845558261444<br>SJ: 36.66558846963 |

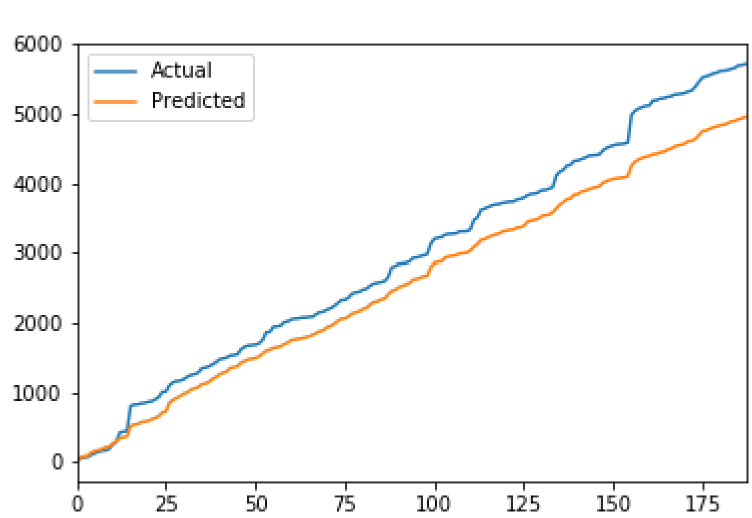
**Gradient Boosting (Lag by 2 weeks):**

Trials:

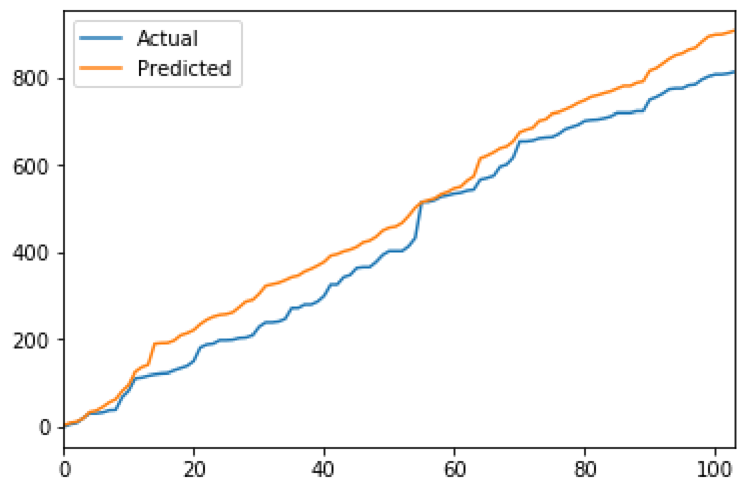
| Classifier | Model | MAE | RMSE |
|---|---|---|---|
| Gradient Boosting | params = {'n_estimators': 500, 'max_depth': 4, 'min_samples_split': 2, 'learning_rate': 0.01, 'loss': 'ls'} | IQ: 4.7572038497669<br>SJ:13.169780296629 | IQ: 7.2294360285830<br>SJ: 20.95923831758 |
| Gradient Boosting | params = {'n_estimators': 500, 'max_depth': 5, 'min_samples_split': 3, 'learning_rate': 0.01, 'loss': 'ls'} | IQ:5.0260450797526<br>SJ:11.408086364235 | IQ:7.463905022157<br>SJ:19.328958358616 |
| Gradient Boosting | params = {'n_estimators': 500, 'max_depth': 4, 'min_samples_split': 2, 'learning_rate': 0.01, 'loss': 'lad'} | IQ:4.8486085606504<br>SJ:13.528954584406 | IQ:8.181314306829<br>SJ:28.620713870435 |
| Gradient Boosting | params = {'n_estimators': 500, 'max_depth': 5, 'min_samples_split': 2, 'learning_rate': 0.01, 'loss': 'lad', 'criterion':'friedman_mse'} | IQ: 4.825438011805<br>SJ:13.067857224752 | IQ:8.044076893817<br>SJ: 27.073608236187 |

**Predicted vs Actual plots:**

1. **Plot for SJ City**




## 2. Plot for IQ City




**Random Forest (Lag by 1 week):**

Trials:

| Classifier | Model | MAE | RMSE |
|---|---|---|---|
| Random Forest | `regr = RandomForestRegressor(max_depth=2, random_state=0)` | IQ:5.4241449298766<br>SJ:25.012271229010 | IQ: 7.604797919619<br>SJ: 44.95453056896 |
| Random Forest | `regr = RandomForestRegressor(max_depth=3, random_state=0)` | IQ:5.235411128368<br>SJ:19.72014768472 | IQ: 8.416833419060<br>SJ: 33.24496148916 |
| Random Forest | regr = RandomForestRegressor(max_depth=3, random_state=0, max_features='log2') | IQ:5.4960218678848 | IQ: 8.484419118123<br>SJ: 49.44246328686 |

| | | | |
|---|---|---|---|
| | | SJ: 26.642047723406 | |
| Random Forest | p regr = RandomForestRegressor(max_depth=10, random_state=1, max_features='log2',min_samples_split=3) | IQ:5.4008849722292<br>SJ: 23.9122358438 | IQ: 8.738930557529<br>SJ:46.230081560587 |

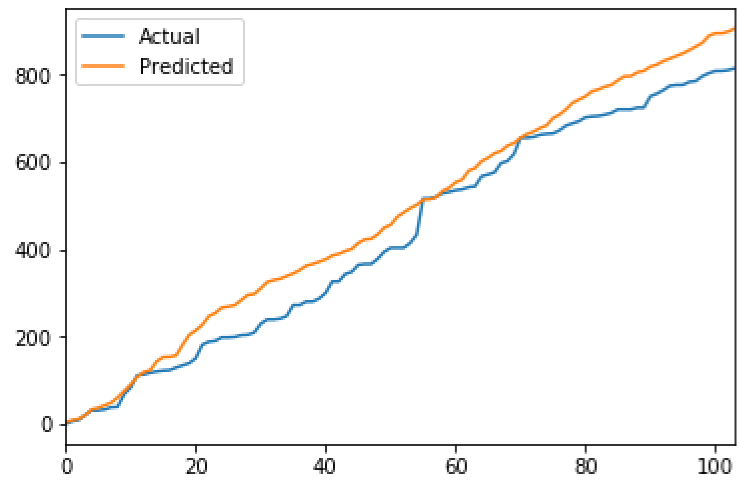
**Random Forest (Lag by 2 weeks):**

Trials:

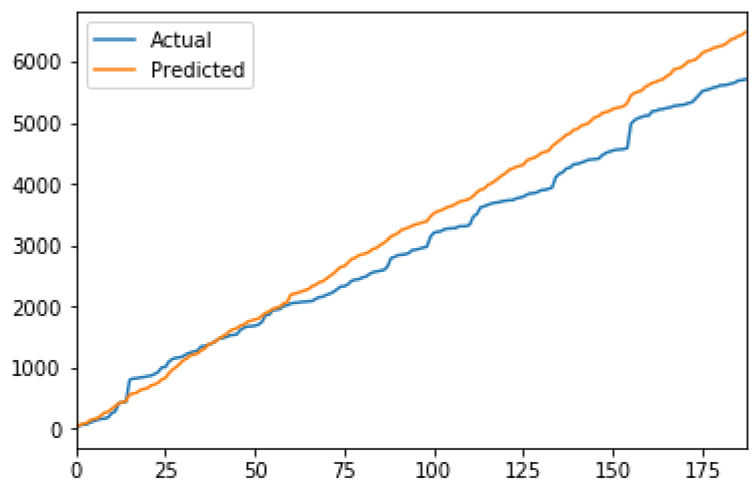
| Classifier | Model | MAE | RMSE |
|---|---|---|---|
| Random Forest | `regr = RandomForestRegressor(max_depth=2, random_state=0)` | IQ:6.100055808380<br>SJ:23.117544192869 | IQ: 8.806959599483<br>SJ:36.35417094833 |
| Random Forest | `regr = RandomForestRegressor(max_depth=3, random_state=0)` | IQ:5.815906583559<br>SJ:18.50039516485 | IQ: 8.425190810639<br>SJ:31.107331513269 |
| Random Forest | regr = RandomForestRegressor(max_depth=3, random_state=0, max_features='log2') | IQ:6.126982632018<br>SJ:22.97877972265 | IQ:8.689771924678<br>SJ:37.402646854778 |
| Random Forest | p regr = RandomForestRegressor(max_depth=10, random_state=1, max_features='log2',min_samples_split=3) | IQ:6.212865103418<br>SJ:19.68870702445 | IQ: 8.959740137273<br>SJ:32.70381714878 |

**Predicted vs Actual Cases Plot:**

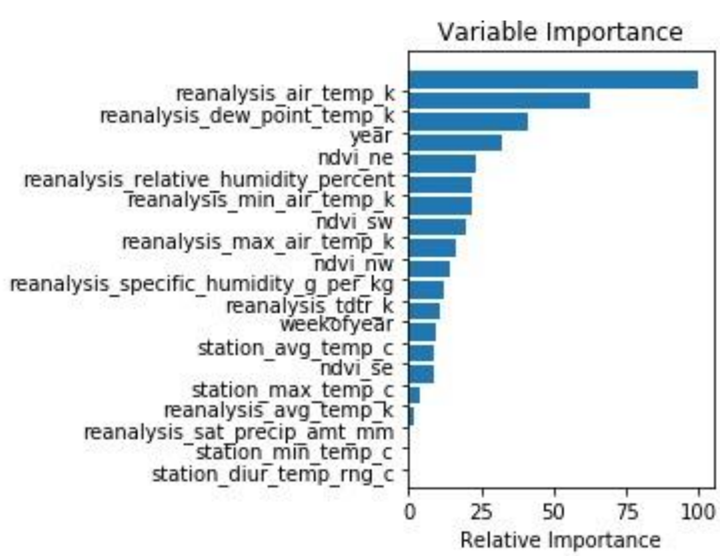
1. **Plot for SJ City**




2. **Plot for IQ City**




3. **Variable Importance**



# Conclusion:

From our approach where we considered the two cities Iquitos, located in Peru, South America and San Juan, the capital of Puerto Rico. Based on the domain knowledge we identified that these two cities are located on mildly different geographical locations. Therefore, our approach was also designed such that we consider these two cities independently. Also, our very vital assumption for this domain that we identified was that it takes 5-7 days for a human to get affected by the dengue fever. Hence, we have shifted the total affected cases by 1 week and 2 weeks in order to observe the trend.

We have applied various classifiers after scaling the dataset and careful omission of highly co-related attributes and other attributes that do not enhance the data conversion to information.

From our observations, we were able to achieve best results by using Gradient boosting, an ensemble technique. Our competition required us to calculate the results in terms of Mean Absolute Error(MAE) and we were able to achieve a value of 4.2687605543857 for the city of Iquitos and 11.408086364235 for the city of San Juan.

Gradient boosting, a class of Ensemble technique turned out to help us build a strong predictive model as it uses multiple weak learners by the concept of additive model and tries to reduce the loss function. In this case the square error value as this is a regression problem. Therefore, even though we have used multiple classifiers such as Random Forests, K-NN, SVM and Deep Learning along with Gradient Boosting. We identified Gradient boosting to be the ideal classifier.

# Contributions:

Christopher Kassap: Assisted with model evaluation, and wrote the report based on test results.

Vutukuru, Ram Anand: Assisted with model evaluation, and tested the model against gradient boosting and random forest classifiers.

Kataria, Jaminee: Assisted with model evaluation, and tested the model against K-NN and SVM classifiers.

Kaneria, Dwaniben Rameshbhai: Assisted with model evaluation, pre-processing, and tested the model against Neural Network classifiers.