KLE Society's
KLE Technological University, Hubballi.

A Machine Learning(22ECSC306) Course Project Report

On

DengAI

submitted in partial fulfillment of the requirement for the degree of

Bachelor of Engineering

In

School of Computer Science and Engineering

Submitted By

Praveen Kulli	01fe20bcs012
Ananya Patil	01fe20bcs017
Chirag Sandeep Metgud	01fe20bcs018
Anurag Joshi	01fe20bcs030

Under the guidance of
Mrs. Sneha Varur

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

HUBBALLI – 580 031

Academic year 2022-23

KLE Society's
KLE Technological University, Hubballi.

2022 - 2023

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that Machine Learning (22ECSC306) Course Project entitled DengAI is a bonafied work carried out by the student team Praveen Kulli 01fe20bcs012, Ananya Patil 01fe20bcs017, Chirag Sandeep Metgud 01fe20bcs018, Anurag Joshi 01fe20bcs030 in partial fulfillment of completion of Vth semester B. E. in Computer Science and Engineering during the year 2022-2023. The course project report has been approved as it satisfies the academic requirement with respect to the course project work prescribed for the above said program.

Guide

Mrs. Sneha Varur

Head,
School of Computer Science and Engineering
Dr. Meena S. M

External Viva -Voce:

Name of the Examiners	Signature with date
1.	
2.	

ABSTRACT

Dengue is caused by four different types of viruses, most frequently the Aedes aegypti mosquito which spread dengue illness. In its milder form, dengue causes moderate to severe flu-like symptoms in infected individuals, but severe dengue, also known as dengue hemorrhagic fever, can be fatal if untreated quickly. Millions of individuals living in tropical and subtropical regions of the world, where the mosquito thrives, are thought to be seriously at risk for the disease. Numerous studies have shown a favorable relationship between climatic factors, notably temperature, humidity, and precipitation levels, and the occurrence of dengue disease. Many of these studies include quantitative models that re- late climate factors to the occurrence of dengue cases. The mathematical models make us wonder how well we will be able to predict illness occurrences in the future using climate variables included in weather forecasts. The Dengue Forecasting project, which makes climate and dengue data available to data scientists at large and challenges them to submit prediction models to help anticipate future dengue epidemics, was started in response to this question by many departments of the US Federal Government. In this study, we create a predictive model and assess how they perform against more widely applied machine learning models. It can be seen that the transmission of Dengue fever is related to various climate variables such as temperature and precipitation. Although the relationship between the transmission of Dengue with climate variables is complex, a number of experts argue that climate change is likely to produce a great effect on the spread of Dengue fever. The problem we are trying to solve is, to predict the number of dengue cases (number of patients who are caught with Dengue) each week (in each location) based on environmental variables describing changes in temperature, precipitation, vegetation, etc. The competition provides a data set containing around 1450 entries, of climate conditions and reported dengue cases in two cities; San Juan and Iquitos. The goal is to predict the total cases label for each (city, year, week of year) in the test set. The objective of the competition Denga AI is to predict a specific number of dengue cases in a specific length of time. Each era incorporates weather information from many sources. Each participant receives a training and testing dataset (without target values). After submission, the score is determined by calculating the MAE (Mean Absolute Error) between model predictions and true outcomes.

Keywords: Dengue; climatic conditions

ACKNOWLEDGEMENTS

The sense of contentment and elation that accompanies the successful completion of our project and its report would be incomplete without mentioning the names of the people who helped us in accomplishing this.

We are indebted to our guide Mrs.Sneha Varur who was nothing but a constant source of enthusiasm and whose profound guidance, valuable suggestions, and beneficent direction was mainly responsible for us to complete this project.

We take this opportunity to express our deep sense of gratitude and sincere thanks to our Head , School of Computer Science and Engineering Dr. Meena S. M. for her tremendous source of inspiration and help in challenging our effort in the right direction.

Last but not least we like to thank all the course faculty, teaching and non-teaching staff for helping us during the project.

Praveen Kulli
Ananya Patil
Chirag Metgud
Anurag Joshi

Contents

1 INTRODUCTION

Figure 1: Aedes aegypti

Aedes aegypti mosquitoes, the most common carrier of the four related kinds of dengue viruses, are responsible for the sickness. Although mild dengue, also known as dengue hemorrhagic fever, can be fatal if untreated, infected patients will have mild to severe flu-like symptoms in its more severe form. According to the Dengue and Severe Dengue Fact Sheet published by the World Health Organization (WHO), "severe dengue affects most Asian and Latin American countries and has become a leading cause of hospitalisation and death among children and adults in these regions." "Dengue incidence is widespread in tropical and subtropical regions of the world. Not only does the primary disease vector, the Aedis aegypti mosquito, thrive in warm, humid climates with high levels of precipitation, but warm temperatures also worsen the disease by speeding up viral replication in the mosquito. According to the Centers for Disease Control and Prevention's (CDC) "Dengue and Climate" website, abrupt changes in the weather, particularly those that affect temperature, precipitation, and humidity, are frequently linked to an increase in dengue incidence in countries where transmission does happen frequently. In order to construct prediction models to forecast dengue using climate-related data, the US Department of Commerce initiated the Dengue Forecasting Initiative in 2015. Numerous academics have investigated and validated this association between climate and this disease. With an emphasis on neural network models, we will investigate this data in our research and develop original predictive models. We will enter our findings into the DrivenData competition "DengAI: Predicting Disease Spread" to contrast the performance of our model, particularly the performance of our neural net models, with that of other data scientists. Dengue morbidity can be decreased by deploying improved epidemic prediction and detection methods, according to the WHO's article "Global plan for dengue prevention and control 2012-2020." One of the numerous techniques to assess

the danger of upcoming Dengue outbreaks is the use of models that statistically estimate the incidence of the disease based on climate data.

1.1 All about Drivendata Challenge

Making forecasts regarding the weekly frequency of dengue cases (in each area) based on environmental elements like temperature changes, precipitation, vegetation changes, and more is required for the competition in the healthcare sector. Severe dengue is indicated by ruptured and leaky blood vessels. Additionally, platelets—cells that aid in blood clotting—decrease in number in the blood. This can lead to internal bleeding, shock, organ failure, and even death. Dengue is spread by Aedes mosquitos, which are also responsible for spreading the illnesses Zika and chikungunya. When a mosquito bites someone who has dengue fever, it spreads the disease to the subsequent victim.

The contest is hosted on Driven data Platform
Deadline: April 7, 2023 - Final submission deadline

1.2 Motivation

Only tropical and subtropical regions of the world are susceptible to dengue fever, a deadly viral disease spread by mosquitoes. Climate variables including temperature, precipitation, humidity, and others have an impact on the transmission of dengue, a virus spread by mosquitoes. As a result, by using the correlation between these environmental parameters and past data, it is possible to predict dengue cases.Dengue fever has been spreading over the years and has become a major problem . Over 0.5 billion people per year in Latin America has been infected due to it . A complex task to model the relation between climate and dengue dynamics can improve research initiatives and resource allocation to help fight life-threatening pandemics.

1.3 Objectives

The objectives of this project are:

(1) Create a model capable of producing competitive results when compared to results obtained by other data scientists and aspiring data scientists participating in the DrivenData competition "DengAI: Predicting Disease Spread".

(2) Do so by using multiple models and selecting the optimum model

1.4 Literature Survey

[1] Predictive Modeling of Dengue Fever Epidemics: A Neural Network Approach

Author: Carlos Sathler
Published on: December 10, 2017

According to the CDC, a third of the world's population may be exposed to the dengue virus and at risk of getting sick. 11 The disease affects 400 million individuals worldwide annually, killing 25,000 people. 12 Meanwhile, the WHO claims that "better epidemic prediction and detection can minimise dengue morbidity." 9 Therefore, developing accurate predictive models is crucial for containing this condition. In our project, we developed dengue epidemic forecasting models in conjunction with the "DengAI: Predicting Disease Spread" competition from DrivenData8 and the US Department of Commerce Dengue Forecasting project7.We compared our findings with those of more widely used machine learning techniques, such as decision tree regression and various multivariate linear regression models, in order to predict dengue cases from climatic data. We entered our findings into the DengAI competition and discovered that the Multi-Layer Perceptron, LongShort Term Memory, and Gated Recurrent Unit networks performed worse than more conventional machine learning techniques. Despite the fact that our results fell short of the competition, we still feel that our effort has been worthwhile thanks to the careful evaluation and documenting of many neural network techniques to dengue prediction. If more complex neural network architectures (like seq2seq and UFCNN) are capable of providing better, more competitive results.

[2] DengAI: Predicting Disease Spread Inductive Methods of Data Analysis

Author: Kemal Erdem
Published on: June 2, 2020

Tropical and subtropical regions of the world are affected by dengue fever, a mosquito-borne illness. Flu-like symptoms, such as fever, rash, and joint and muscular discomfort, can occur in moderate instances. Dengue fever may, in extreme situations, result in fatal haemorrhage, low blood pressure, and other complications. Dengue's dynamics of transmission are influenced by climate factors including temperature and precipitation because it is a mosquito-borne disease. Though the connection to climate is nuanced, an increasing number of scientists contend that global distributional shifts brought on by climate change are likely to have significant effects on public health. Its goal is to predict the overall number of dengue fever cases that will be reported each week in San Juan, Puerto Rico, and Iquitos, Peru. We receive environmental data gathered by various federal government agencies in the United States, including the Centers for Disease Control and Prevention, the National Oceanic and Atmospheric Administration, and others.

[3] Knowledge, attitude and practice regarding dengue infection among Jazan inhabitants, Saudi Arabia 2019

Authors: Osama B Albasheer, Amani Osman Abdelmola, Abeer Alomaish, Alanood Dallak, Kawakeb Darraj, Marwah hamzi, Hanan Shawlan, Shareefa Sumaily, Marwah Gomairy, Rana hakami,Mohammed S Mahfouze

Published on: March 2021

The people of Jazan are adequately informed on dengue disease transmission, manifestation, and severity. However, they have little experience guarding against mosquito bites. Therefore, it is advised that community-based educational programmes with a focus on social mobilisation and awareness-raising should be developed in order to transform knowledge into sound practice.

[4] Dengue virus: A global human threat: Review of literature

Authors: Shamimul Hasan, Sami Faisal Jamdar, Munther Alalowi

With an estimated 2.5 billion victims spread over more than 100 nations, dengue has developed into a serious public health hazard. In order to guarantee an early and appropriate treatment strategy, the doctor should be informed of the many clinical signs of this illness. The development of a vaccine, mosquito control strategies, and antiviral medication regimen are the future approaches in the fight against this horrible illness.

2 PROPOSED SYSTEM

2.1 Understanding the Data

Predicting the total-cases label for each test set variable (city, year, week of year) is the objective. The test data for the two cities, San Juan and Iquitos, span 5 and 3 years, respectively. Predictions for both cities are included in each entry. Each city's data has been combined with a city column designating the source, such as sj for San Juan and iq for Iquitos. Since there is no overlap between the test data and any of the training data, the test set is a pure-future hold-out. Missing values have been replaced throughout using NaNs.

Data analysis We integrated the features from the training and test data sets in order to analyse the feature set. When the statistics were first examined, it became clear that Iquitos and San Juan had somewhat different characteristics. Most notably, there were vastly different data in the total cases (the target variable). This might be as a result of the towns' population (density). As a result, we choose to divide the data into different cities.

2.2 Data set description

The Dataset consists of 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 Daily climate data 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

Satellite precipitation measurements:

• precipitation-amt-mm : Total precipitation

Climate Forecast System Reanalysis measurements:

- 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

- 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 input and submission format For example, a single row in the dataset, indexed by (city, year, weekofyear): (sj, 1994, 18), has these values: Input data sample:

week-start-date 1994-05-07

total-cases 22

station-max-temp-c 33.3

station-avg-temp-c 27.7571428571

station-precip-mm 10.5

station-min-temp-c 22.8

station-diur-temp-rng-c 7.7

precipitation-amt-mm 68.0

reanalysis-sat-precip-amt-mm 68.0

reanalysis-dew-point-temp-k 295.235714286

reanalysis-air-temp-k 298.927142857

reanalysis-relative-humidity-percent 80.3528571429

reanalysis-specific-humidity-g-per-kg 16.6214285714

reanalysis-precip-amt-kg-per-m2 14.1

reanalysis-max-air-temp-k 301.1

reanalysis-min-air-temp-k 297.0

reanalysis-avg-temp-k 299.092857143

reanalysis-tdtr-k 2.67142857143

ndvi-location-1 0.1644143

ndvi-location-2 0.0652

ndvi-location-3 0.1321429

ndvi-location-4 0.08175

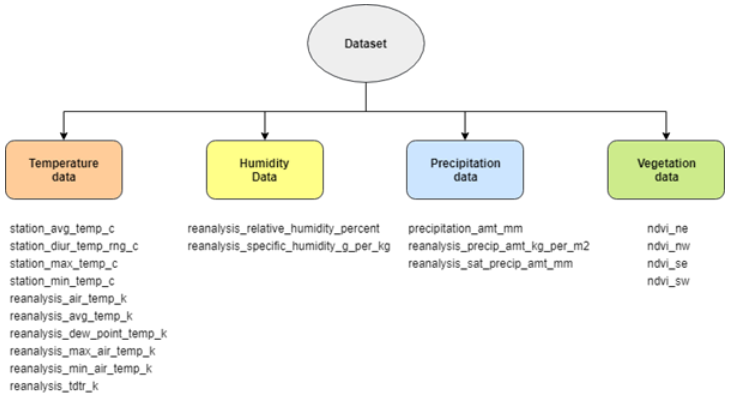


Figure 2: Dataset Description

Submission format:

city,year, weekofyear,total-cases

sj,1990,18,4

sj,1990,19,5

...

Score calculation The Mean Absolute Error is used to evaluate performance (MAE)

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n}$$

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1456 entries, 0 to 1455
Data columns (total 24 columns):
 #   Column                                 Non-Null Count  Dtype
---  ------                                 --------------  -----
 0   city                                   1456 non-null   object
 1   year                                   1456 non-null   int64
 2   weekofyear                             1456 non-null   int64
 3   week_start_date                        1456 non-null   object
 4   ndvi_ne                                1262 non-null   float64
 5   ndvi_nw                                1404 non-null   float64
 6   ndvi_se                                1434 non-null   float64
 7   ndvi_sw                                1434 non-null   float64
 8   precipitation_amt_mm                   1443 non-null   float64
 9   reanalysis_air_temp_k                  1446 non-null   float64
 10  reanalysis_avg_temp_k                  1446 non-null   float64
 11  reanalysis_dew_point_temp_k            1446 non-null   float64
 12  reanalysis_max_air_temp_k              1446 non-null   float64
 13  reanalysis_min_air_temp_k              1446 non-null   float64
 14  reanalysis_precip_amt_kg_per_m2        1446 non-null   float64
 15  reanalysis_relative_humidity_percent   1446 non-null   float64
 16  reanalysis_sat_precip_amt_mm           1443 non-null   float64
 17  reanalysis_specific_humidity_g_per_kg  1446 non-null   float64
 18  reanalysis_tdtr_k                      1446 non-null   float64
 19  station_avg_temp_c                     1413 non-null   float64
 20  station_diur_temp_rng_c                1413 non-null   float64
 21  station_max_temp_c                     1436 non-null   float64
 22  station_min_temp_c                     1442 non-null   float64
 23  station_precip_mm                      1434 non-null   float64
dtypes: float64(20), int64(2), object(2)
memory usage: 273.1+ KB
```

Figure 3: Features Train Data.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 416 entries, 0 to 415
Data columns (total 24 columns):
 #   Column                                 Non-Null Count  Dtype
---  ------                                 --------------  -----
 0   city                                   416 non-null    object
 1   year                                   416 non-null    int64
 2   weekofyear                             416 non-null    int64
 3   week_start_date                        416 non-null    object
 4   ndvi_ne                                373 non-null    float64
 5   ndvi_nw                                405 non-null    float64
 6   ndvi_se                                415 non-null    float64
 7   ndvi_sw                                415 non-null    float64
 8   precipitation_amt_mm                   414 non-null    float64
 9   reanalysis_air_temp_k                  414 non-null    float64
 10  reanalysis_avg_temp_k                  414 non-null    float64
 11  reanalysis_dew_point_temp_k            414 non-null    float64
 12  reanalysis_max_air_temp_k              414 non-null    float64
 13  reanalysis_min_air_temp_k              414 non-null    float64
 14  reanalysis_precip_amt_kg_per_m2        414 non-null    float64
 15  reanalysis_relative_humidity_percent   414 non-null    float64
 16  reanalysis_sat_precip_amt_mm           414 non-null    float64
 17  reanalysis_specific_humidity_g_per_kg  414 non-null    float64
 18  reanalysis_tdtr_k                      414 non-null    float64
 19  station_avg_temp_c                     404 non-null    float64
 20  station_diur_temp_rng_c                404 non-null    float64
 21  station_max_temp_c                     413 non-null    float64
 22  station_min_temp_c                     407 non-null    float64
 23  station_precip_mm                      411 non-null    float64
dtypes: float64(20), int64(2), object(2)
memory usage: 78.1+ KB
```

Figure 4: Features Test Data.

2.3 Data Pre-processing

In the training data we use forward filling to fill the missing values. Rather than creating new Dataframe we modified it in place. We filled the missing values with mean.We splitted the city column into two new data frames San Jaun and Iquitos. Features changed over time: The index of both cities data frame are set to date. The week start date column will be used as the index for each cities. Week start date column is removed from data frame after it is used to set index. The new column is created for both the cities known as ndvi mean. The values in this column is calculated by taking mean of values in a subset column.Standardizing the values of two cities San juan and iquitos using the standard scalar class. Standardization is common pre-processing step that scales the values in the dataset so that they have the mean of 0 and standard deviation of 1. This is usefull for ML algorithms that are sensative to scale of the input features.The resulting standardized values are stored in df-sj-s, the same operation is performed on city Iquitos. We have set the index to dates using the column week start date for both san juan and Iquitos city. New column ndvi mean created for each week for both city.By the plots we realized The NVDI scores in the Southwest and Southeast are consistently lower than the scores of the Northwest and Northeast quadrants.In addition, we can see the impact of the front fill method for data imputation. In 1995, there were a few missing rows of NVDI data. The flat line comes from repeating the last known value over and over until an observation was

recorded again in the city of san Juan. Weather: In order to visualize weather features of various metrics, we standardize both the cities, to get rid of the units , Millimeters of rain and degrees Celcius have been choosen. We normalized both the weather features of cities.

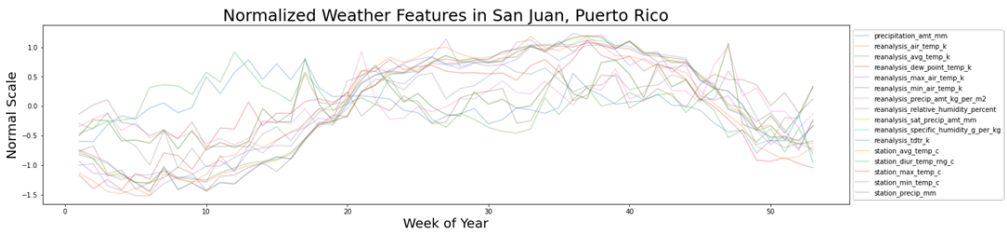


Figure 5: Normalized weather Features in San Juan, Puerto Rico

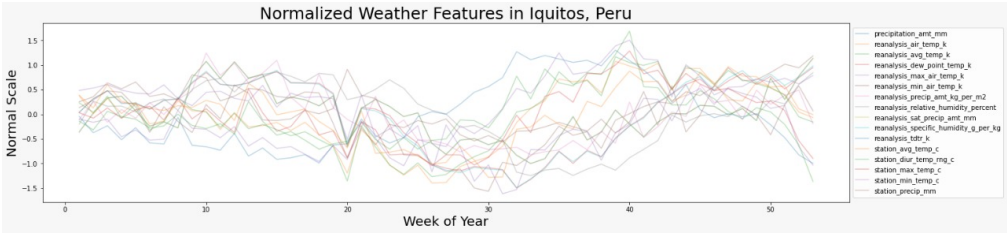


Figure 6: Normalized weather Features in Iquitos,Peru

2.4 Initial Approach

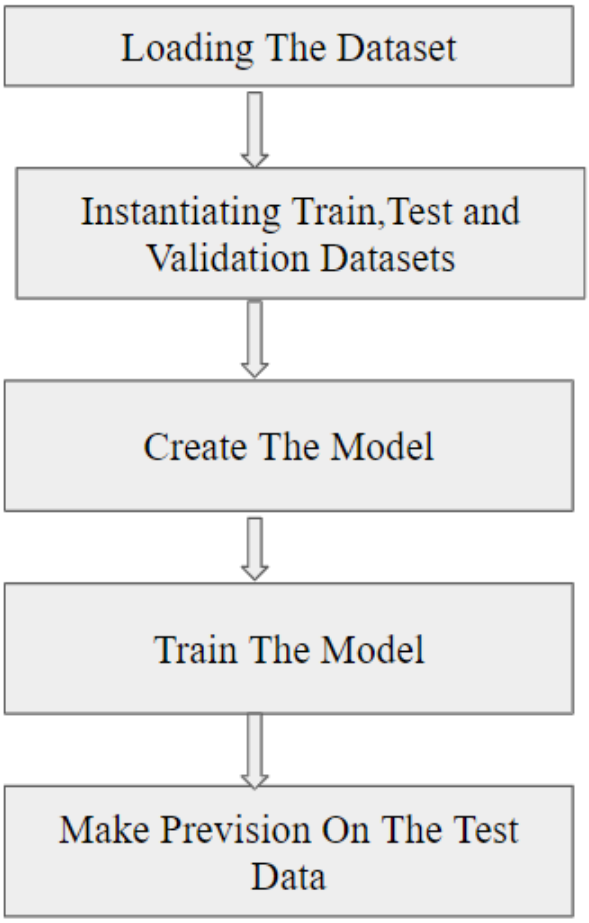


Figure 7: Flow process for methodology

2.5 Solution Procedure

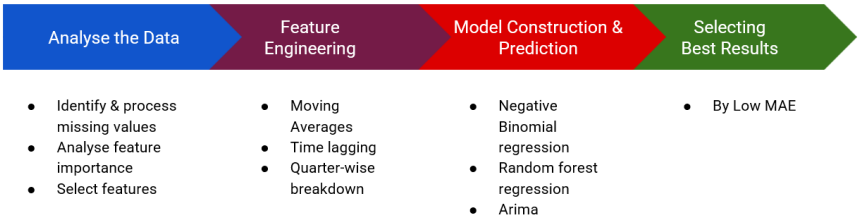


Figure 8: Solution Procedure

2.6 Proposed methodology

Random Forest Regressor
Random Forest Regressor is an ensemble model that combines numerous decision trees' predictions to produce a prediction that is more accurate than any single tree could be.Each decision tree in a random forest regressor is trained on a particular subset of data to produce a forecast for a specific input. The forecasts of all the trees are then integrated to create a final prediction. Typically, this is done by averaging all of the predictions.The ability of a random forest regressor to handle high-dimensional data, such as data with numer- ous characteristics or variables, is one of the main benefits of utilising one.Additionally, compared to other kinds of regression models, random forests are frequently fairly accu-rate.Overall, a random forest regressor is an effective tool for doing regression tasks that can yield precise predictions even when using complicated or highly dimensional data.

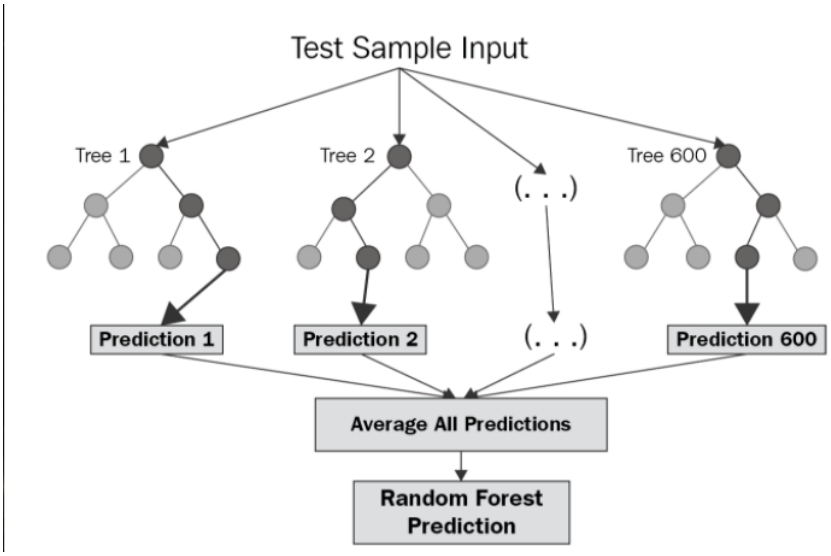


Figure 9: Random Forest Regressor

Negative binomial Regressor
Count data are modelled using negative binomial regression, a form of regression analysis. Count data is a sort of data that shows how frequently an event occurs, such as how many people have a particular condition or how frequently an event happens. Negative binomial regression can model overdispersion, which is when the data show more variability than would be anticipated based on the Poisson distribution. It is related to Poisson regression, another technique used to analyse count data. Because of this, negative binomial regression can be used to examine count data that exhibits overdispersion.

Arima Ananlysis

The statistical model known as ARIMA, or "AutoRegressive Integrated Moving Average," is one that is used for time series analysis. Data that is gathered over time is known as a time series, and examples include monthly sales figures and daily temperature observations. By considering previous values in the series and the relationships between those values, ARIMA models are used to analyse and make predictions about time series data.The moving average (MA), the differencing component, and the autoregressive (AR) component are the three parts that make up an ARIMA model.The relationship between a current value in the series and previous values in the series is modelled using the AR component. The time series is made stationary using the differencing component, which ensures that the series' statistical characteristics remain constant across time. The relationship between the residuals—the discrepancy between the observed values and the anticipated values—and earlier residuals is modelled using the MA component.An ARIMA model may efficiently evaluate and predict time series data by combining these three elements.

3 IMPLEMENTATION

3.1 Random forest

In the model Random Forest regression ,to calculate correlations for the original ndvi se column, a rolling mean version of the ndvi se column, and shifted version of the rolling mean column. The rolling method is used to calculate the rolling mean of the ndvi se column with the window size of 27, this means that the mean of 27 values prior to each value in the column will be calculated and stored in moving.The shift method is then used to shift the values in the moving DataFrame by 40.This means that each value in the moving DataFrame will be replaced by the value that is 40 positions.In the pre-processing of the city San Juan several features are selected. Shifting of reanalysis saturation precipitation amount mm by 20 rows and replaced in another dataframe. The other selected features computes a moving average using windows, it is used to smooth out the data by taking the average of set of data points over a certain period. Several operations on the data like filling missing values, converting the week start date column to date time data type adding new column that encode the quarter of the year for each data point, by dropping the original week start date column. The dataframe is divided into two data frames of both the cities. And similar pre-processing is done for Iquitos city. Split validation and training: The train test split function splits the sj train and iq train into training and testing sets. sj train and iq train: The dataframes to be split into training and testing sets. sj label['total cases'] and iq label['total cases']: The labels for the data. test size: The code sets the test size to 10 random-state: The seed for the random number generator used to perform the split. The code sets this to 1 to ensure that the same split is generated every time the code is run. Train Model: The model is using random forest regression to make prediction on two seperate dataset sj and iq. The models are first created and specified with certain parameters, such as the number of estimators and the criterion used for evaluation. The models are then trained on respective training dataset once trained the models are used to make prediction and the performance of model is evaluated using mean absolute error.

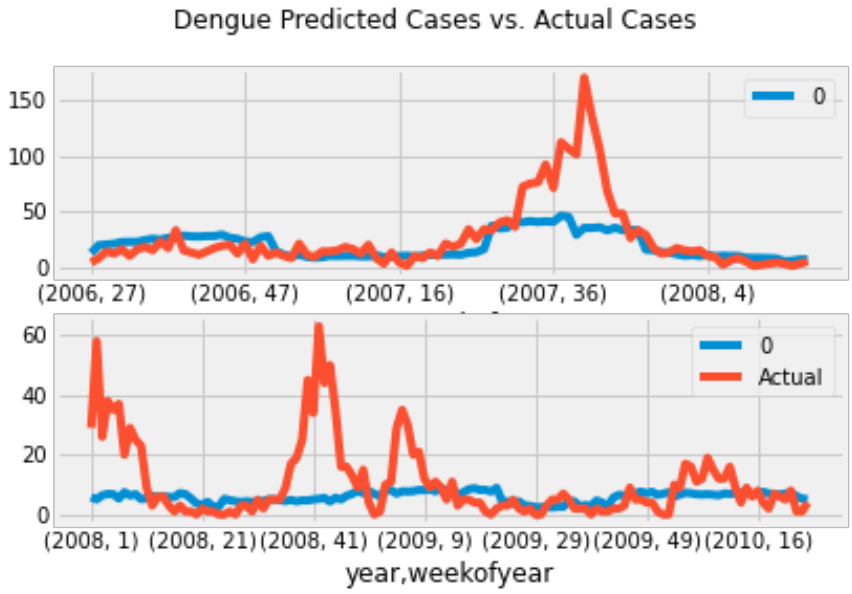


Figure 10: Dengue Predicted Cases Vs. Actual Cases

Plot Prediction: Figure with two subplots, one for each of the sj and iq datasets. The first subplot is populated with a line plot of the predicted values from the model (stored in the sj pred val variable) and the actual values from the test set (stored in the sj test y variable). The second subplot is populated with a line plot of the predicted values from the model (stored in the iq pred val variable) and the actual values from the test set (stored in the iq test y variable). Save prediction: Makes predictions on the test data using the previously trained sj model and iq model models. Creates two Pandas data frames, one for each of the sj and iq datasets, to hold the predictions and their rolling average. Loads the submission, Replaces the total cases column of the submission dataframe with the concatenation of the rolling average predictions from the sj and iq datasets.

3.2 Negative Binomial Regression

In Negative Binomial Regression model the seleceted features has been extracted from feature engineering. Separate-and-engineer is a function that takes in several arguments: city, data, selected-features, labels, mv1, mv2, mv3, mv4.The selected features is used in the model. The first thing the separate- and-engineer function does is convert the week-start-date column in the data DataFrame to datetime format using to datetime method.Iteration takes over the numbers 1 through 4, and for each number, it adds a new column to the data DataFrame called quarter-X, where X is the number from the loop. The values in this new column are 1 if the month of the week-start-date falls within

the range of the quarter represented by X, and 0 oth- erwise. For example, if X is 1, the new column will have a value of 1 for dates that fall in the first quarter of the year (January, February, and March), and 0 for all other dates. The selected-features list is then updated to include the new quarter-X columns.the data include only rows where the city column matches the city argument.It then selects only the columns in selected features from the subsetted DataFrame and assigns the result to a new DataFrame called city-train-features. Finally, the code selects only the numeric columns from city-train-features and assigns the result to city-train-features again. It computes moving averages for several columns in the city-train-features df and adds new columns to the df for these moving averages.The window size for the moving average is specified by the mv1, mv2, mv3, and mv4 arguments, which represent the number of rows to include in the moving average calculation for each of the corresponding columns. Next,the fillna method to fill any remaining missing values in the city-train-features df with the mean value for each column.If the labels argument is not None,then it subsets the labels df to include only rows where the city column matches the city argument, fills any missing values with the column means, and concatenates the resulting df with city- train-features along the columns axis. The resulting df is then returned by the function.If the labels argument is None, the code simply returns the city-train-features df, get-best-model is a function that takes two arguments, train and test. By defining the model-formula variable as a string containing a formula that specifies the model. This

formula appears to use the total number of cases as the dependent variable and several other variables as independent variables. Then it defines a grid variable that contains a range of possible values for the alpha parameter in the model. Then it initializes two empty lists, best-alpha and best-score. These will be used later to store the best alpha value and the best score for the model, respectively. Then it loops through the values in the grid variable, fitting a model to the training data for each value of alpha using the specified model formula. For each model, the code makes predictions on the test data, calculates the mean absolute error between the predictions and the true values, and compares this error to the current best score. If the error for the current model is lower than the current best score, the code updates The best-alpha and best-score variables to reflect the new best alpha and score. This can be used to select the best model and evaluate its performance. It refits the best model found on the entire dataset (the concatenation of the training and test sets) using the best alpha value found. This refitted model is then returned by the function along with the best score. The caller can use this model to make predictions on new data.

A function is defined, that selects the best model for predicting the number of dengue cases in (SJ) and (IQ). The function has several steps. First, the function separate-and-engineer is used to split the data into training and testing sets for SJ and IQ, and then further splits the training sets into sub-training and sub-testing sets. Next, it uses the get-best-model function to fit several models to the sub-training data for SJ and IQ and select the best model for each city based on the model's mean absolute error on the

sub-testing data. Then it checks whether the best model for each city has a lower mean absolute error than the current best model for that city. If so, it updates the best model and score for that city. This can be used to evaluate the performance of the best models for each city.

The best models found for San Juan (SJ) and Iquitos (IQ) to make predictions on the training data for each city, and then plotting the predicted and actual values for each city. First it creates a subplot using the subplots function. Next, it uses the best model for SJ to make predictions on the SJ training data, adds these predictions to the sj-train dataframe as a new column called fitted, and plots the fitted and total-cases columns on the first subplot.Then it does the same for the IQ data, adding the predictions made by the best model for IQ to the iq-train dataframe and plotting these predictions and the actual total-cases on the second subplot. This is used to compare the performance of the best models for each city on the training data.

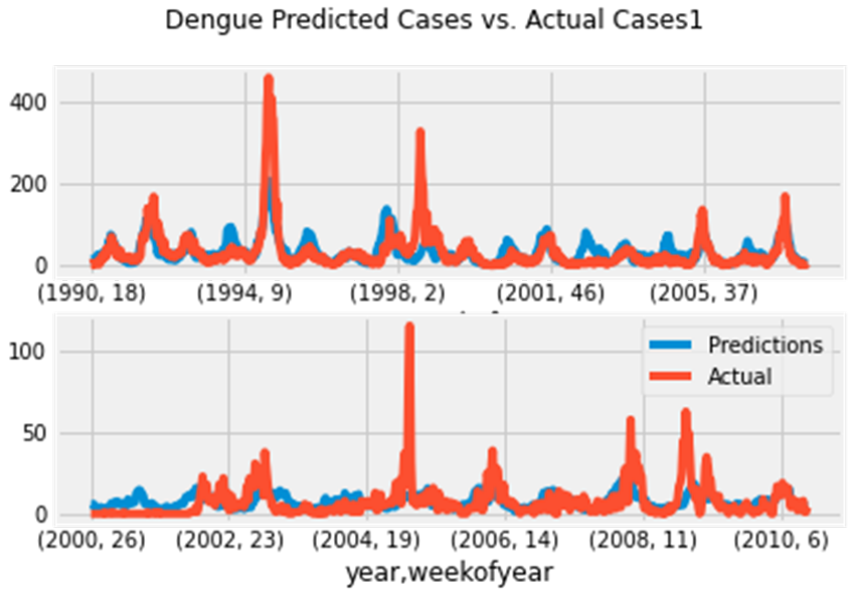


Figure 11: Dengue Predicted Cases Vs Actual Cases

The best models for San Juan (SJ) and Iquitos (IQ) found in the previous section to make predictions on the test data for each city and save these predictions to a file Next, it uses the separate-and-engineer function to split the test data into SJ and IQ dataframes and apply feature engineering to the data. then it uses the best models for SJ and IQ to make predictions on the SJ and IQ test data respectively, and saves these predictions to the variables sj-predictions and iq-predictions.

3.3 Arima Analysis

Arima model is used for Time series data . The index of the DataFrame is set to a Datetime Index using the week start date column of the df-train. Next, two new DataFrame are created, sj-cases and iq-cases, which contain the rows of df-labels that correspond to the cities of 'sj' and 'iq', respectively. To group the data by the start of each month we use resample method, and then sums the total number of cases for each month. The resulting DataFrames are then filled in with missing values using the fillna method and the "backward fill" strategy .

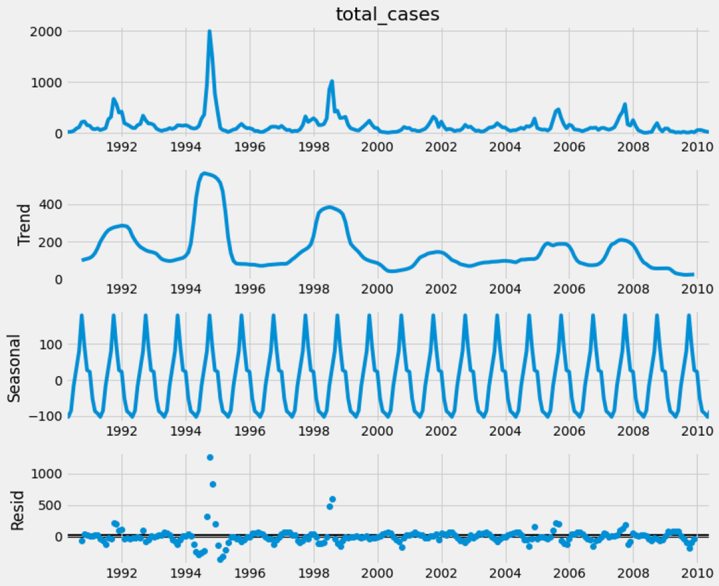


Figure 12: Seasonal Decomposit

A Time series data is plotted and a seasonal decomposition is performed on it. The decomposition separates the data into three components: the trend, the seasonal component, and the residual. The trend, seasonality, and residual components are added together to reconstruct the original time series. The resulting decomposition is then plotted using the plot method. This plot allows us to visualize the individual trend, seasonality, and residual components of the time series data. A combination of parameters are generated for a seasonal ARIMA (SARIMA) model. The p, d, and q parameters, which are set to take any value between 0 and 2. These pa- rameters are used to control the order of the autoregressive (AR) model, the degree of differencing, and the order of the moving average (MA) model, respectively. It then uses the itertools product method to generate all possible combinations of the p, d, and q parameters, and saves them in the pdq list. Similarly, all possible combinations are generated of seasonal p, d, and q parameters, with a fixed seasonal period of 12, and saves them in the seasonal-pdq list.Finally,

it prints out several examples of SARIMA parameter combinations.These examples illustrate how the different combinations of p, d, and q parameters can be used in a SARIMA model. Next it iterating over the different combinations of SARIMA parameters , and using them to train SARIMA models on the sj-cases data.

For each combination of param and param-seasonal from the pdq and seasonal-pdq lists, respectively, it uses the SARIMAX method from the statsmodels library to create a SARIMA model. The sj-cases data is passed as the input to the model, and the order and seasonal-order arguments are set to the current combination of param and param-seasonal values. The model is not required to be stationary or invertible. The fit method is then used to train the SARIMA model, and saves the resulting model in the results variable. Finally, Akaike information criterion (AIC) is printed for the model. The AIC is a measure of the goodness of fit of the model, with a lower AIC indicating a better fit. Overall, it is training a series of SARIMA models on the sj-cases data and out their AIC values. A SARIMA model is trained on the sj-cases data using specific values for the p, d, and q parameters. The SARIMAX method is used to create a SARIMA model, with the sj-cases data passed as the input. The order and seasonal-order arguments are set to specific values of (1,0,1) and (0,1,1,12), respectively, which specify the values of the p, d, and q parameters for the non-seasonal and seasonal components of the model. The enforce-stationarity and enforce-invertibility arguments are set to False as before. The fit method is then used to train the model and saves the results in the results-sj variable. Finally, summary of the model's is produced, including a table of the estimated model parameters. This summary allows us to inspect the model and check its performance.

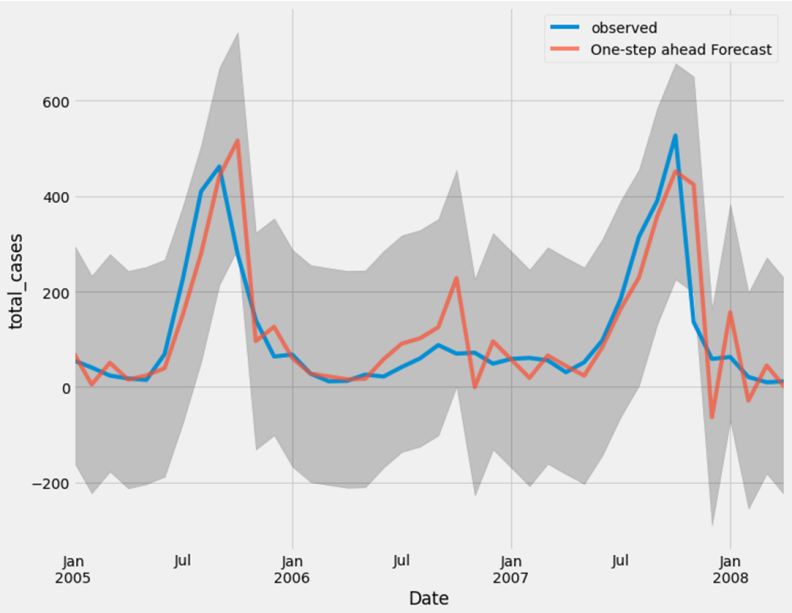


Figure 13: Total number of cases predicted in city San Juan

sj-cases plots data on a graph. The ax variable is being used to set the properties of the graph, such as the x and y axes labels and the legend. The pred-dynamic-sj and pred-dynamic-ci-sj objects are also being plotted on the same graph.

The mean squared error (MSE) is calculated between the predicted values of a time series and the true values of the time series (stored in the sj-cases object for the time period from 2005 to the present). The MSE is a measure of the difference between the predicted and true values, with a lower MSE indicating a better fit. The MSE is calculated by taking the square of the difference between the predicted and true values, and then taking the mean of those squared differences. The get-forecast method is used of the results-sj object to generate 500 steps ahead in the future. The resulting forecast is stored in the pred-uc-sj object. The conf-int method is then used to compute the confidence intervals of the forecasts and store them in the pred-ci-sj object. These confidence intervals provide a range of possible values for the forecasts, with a higher confidence level indicating a narrower range. This can be useful for understanding the uncertainty of the forecasts and determining how likely they are to be accurate. The ax variable is being used to set the properties of the graph, such as the x and y axes labels and the legend. The pred-uc-sj object is also being plotted on the same graph. The fill-between function is being used to add a shaded area to the graph. This plot likely shows the observed values of the time series (from the sj-cases object) as well as the forecasted values (from the pred-uc-sj object), along with the confidence intervals of the forecasts (computed using the pred-ci object). It then performs a seasonal decomposition of the iq-cases time series using the seasonal-decompose function from the statsmodels (sm) library. The decomposition is performed using an additive model, which means that the trend, seasonality, and residual components of the time series are added together to obtain the original time series. The resulting decomposition is stored in the decomposition object. Different combinations of parameters are generated for a Seasonal Autoregressive Integrated Moving Average (SARIMA) model. SARIMA is a time series forecasting method that can be used to model the trend and seasonality of a time series. It uses the itertools.product function to generate all possible combinations of the p, d, and q parameters, which represent the number of autoregressive (AR) terms, the number of differences, and the number of moving average (MA) terms, respectively. The p, d, and q parameters can take any integer value between 0 and 2, so the itertools.product function generates a total of 9 different combinations for each parameter. All possible combinations are generated of seasonal p, d, and q parameters, which are used to model the seasonal component of the time series. The seasonal period (the number of time steps in a season) is set to 12, and the itertools.product function is used to generate all possible combinations of the p, d, and q parameters. Finally, different SARIMA parameter combinations are printed to the output. These combinations are of the form SARIMAX: (p, d, q) x (P, D, Q, S), where (p, d, q) are the non-seasonal parameters and (P, D, Q, S) are the seasonal parameters. The SARIMAX notation indicates that the model includes an exogenous variable (in addition to the time series and its seasonal component).

The SARIMAX function fits a seasonal ARIMA model to the df-labels data. The pdq and seasonal-pdq variables are hyperparameters for the ARIMA model. For each combination of hyperparameters, it fits the model and prints the AIC (Akaike Information Criterion) value. The AIC value is a measure of the goodness of fit of the model, with lower values indicating better fit.It is using a try-except block to handle any errors that may occur when fitting the model with a given set of hyperparameters. Then it is fitting a seasonal autoregressive integrated moving average (SARIMAX) model to the data in the df-labels dataframe. The model has an order of (4,1,3) for the autoregressive (AR), difference (I), and moving average (MA) components, respectively. The seasonal component of the model has an order of (1,1,1,12), where the last number indicates that the seasonality is annual (i.e., there are 12 seasons in the data).The enforce-stationarity and enforce-invertibility arguments are set to False, which means that the model will not be tested for stationarity and invertibility. This can sometimes improve model fit. Once the model is fit, the Akaike information criterion (AIC) is printed, and a summary table of the model's estimated coefficients is also printed. The AIC is a measure of how well the model fits the data and includes a penalty for the number of model parameters, so a lower AIC indicates a better fit. The summary table shows the estimated coefficients and their standard errors, as well as other useful information such as the Wald test and the probability values (p-values).

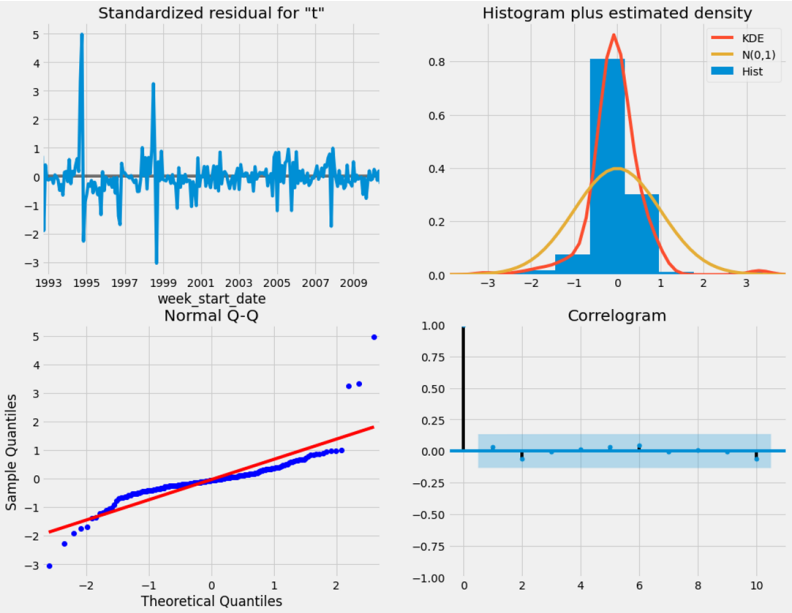


Figure 14: Diagnostics plot

The results.plot-diagnostics method is to be a function that generates diagnostic plots for the results of a statistical model. These plots can help assess the goodness-of-fit of the model and check for things such as heteroscedasticity and normality of residuals.

The results.get-prediction method is used to generate predictions for the statistical model. The start parameter specifies the start date for the predictions, and the dynamic parameter is set to False to indicate that the model should not use the lagged dependent variable in making predictions.The pred.conf-int method is used to generate confidence intervals for the predictions. These intervals provide a range of plausible values for the predictions, taking into account the uncertainty in the model. The resulting confidence intervals can be used to assess the reliability of the predictions.

To generate a plot that compares the observed values of a variable (df-labels) with one-step-ahead forecasts from a statistical model. The ax variable is used to set the plot's x-axis labels to 'Date', and the y-axis labels to 'total-cases'. The observed values are plotted first, followed by the predicted values, which are plotted with a transparent fill to indicate the associated confidence intervals.

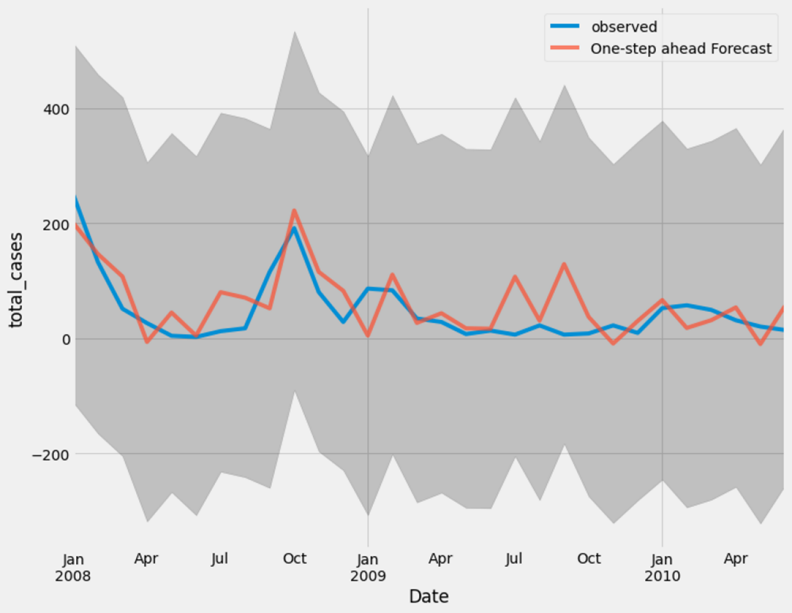


Figure 15: Total number of cases observed

Next ,to calculate the mean squared error (MSE) between the predicted values and the observed values of a variable. The predicted values are stored in the pred.predicted-mean variable, and the observed values are stored in the df-labels dataframe. The MSE is calculated by taking the difference between the predicted and observed values, squaring the result, and then taking the mean of the squared errors.The MSE is a measure of the average squared difference between the predicted and observed values, and can be used to assess the performance of the statistical model.

To generate dynamic predictions using the results of a statistical model. The results.get-prediction method is used to generate the predictions, with the start parameter set to

the date to begin making predictions, the dynamic parameter set to True to indicate that the model should use the lagged dependent variable in making predictions, and the full-results parameter set to True to include the lagged dependent variable in the returned results.The pred-dynamic.conf-int method is then used to generate confidence intervals for the dynamic predictions. These intervals provide a range of plausible values for the predictions, taking into account the uncertainty in the model. The resulting confidence intervals can be used to assess the reliability of the predictions.

4 RESULTS AND DISCUSSION

Regression method and time series method used for generating the dengue prediction model.Mean Absolute Error of Negative Binomial and Random Forest models is comparatively different when considered separately. But MAE of Random forest regressor is less compared to other models. Another model created using time series analysis. Negative Binomial and Random Forest have achieved performance of 28.2813 and 18.1659 MAE respectively and Arima Model got High mean absolute error.By comparing the result models from the regression and time series methods, it is found that the accuracy of time series method is considerably low. Overall Random Forest regressor model has abled to provide better results than other models.This report includes all models used to get a score of 18.0601 MAE and a ranking of 149/12574 in the competition.

4.1 Models used

Model	Methodology	Mean Absolute Error (MAE)
1	Negative Binomial regression	28.2813
2	Random Forest Regressor	18.0601
3	Arima time analysis	High MAE

Figure 16: Table

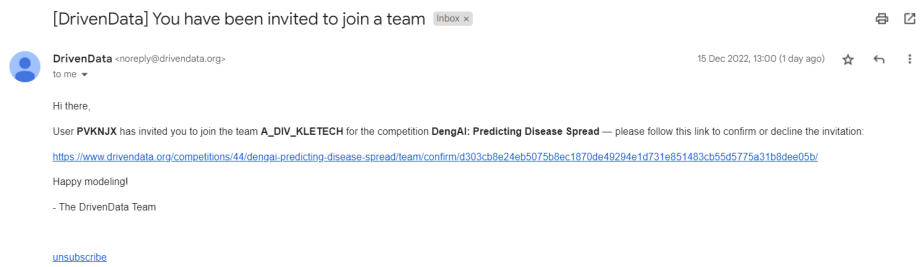
4.2 Leaderboard

Figure 17: Leaderboard

CONCLUSION

To summarize the important points that need to state on this project. Firstly, the model could identify the importance of the preprocessing step. After experimentation, the best way to fill missing values is to fill with the mean of the column when there are no redundant features. Forward and Backward Fill methods were used. In the training data forward filling method to fill the missing values. Moving averages method was used to smooth out the data set. When it comes to feature selection, analyzing the correlation matrix containing correlation between each feature including the total cases. Using the time data find the quarter of data point and add as new feature. Next for model construction and prediction we used Regression method and time series method. By comparing the result models from the regression and time series methods, it is found that the accuracy of time series method is considerably low. In regression method we used two model's Negative binomial and Random Forest regressor. Mean Absolute Error of these two models is different. Negative binomial regression has more mean absolute error (28.2813) compared to Random Forest Regression (18.0601). Making the Random Forest Regression as the best model. So, it is always better to try few models rather than relying on one model. Currently, the rank obtained through the best model (Random Forest regression) is 149 out of 12574 competitors.

4.3 Team


[DrivenData] You have been invited to join a team Inbox ×

DrivenData <noreply@drivendata.org> 15 Dec 2022, 13:00 (1 day ago)
to me

Hi there,

User **PVKNJX** has invited you to join the team **A_DIV_KLETECH** for the competition **DengAI: Predicting Disease Spread** — please follow this link to confirm or decline the invitation:

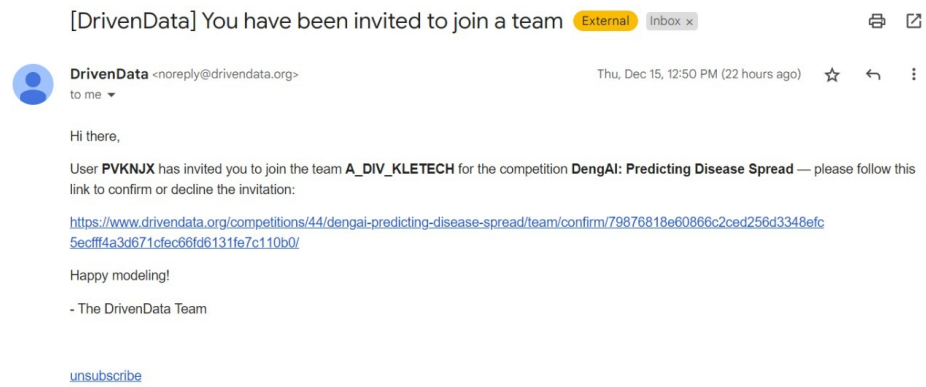
https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/team/confirm/d303cb8e24eb5075b8ec1870de49294e1d731e851483cb55d5775a31b8dee05b/

Happy modeling!

- The DrivenData Team

unsubscribe

Figure 18: Team Member 1


[DrivenData] You have been invited to join a team External Inbox ×

DrivenData <noreply@drivendata.org> Thu, Dec 15, 12:50 PM (22 hours ago)
to me

Hi there,

User **PVKNJX** has invited you to join the team **A_DIV_KLETECH** for the competition **DengAI: Predicting Disease Spread** — please follow this link to confirm or decline the invitation:

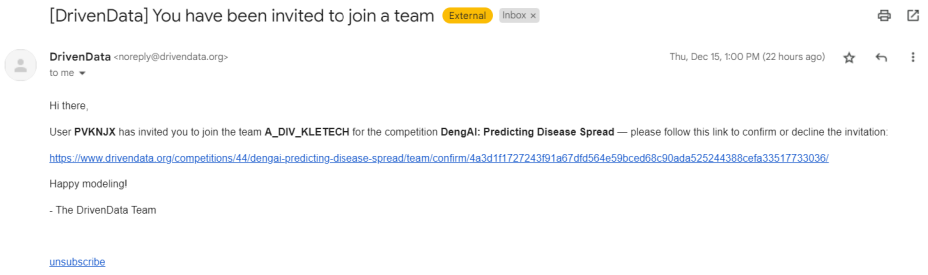
https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/team/confirm/79876818e60866c2ced256d3348efc5ecfff4a3d671cfec66fd6131fe7c110b0/

Happy modeling!

- The DrivenData Team

unsubscribe

Figure 19: Team Member 2


[DrivenData] You have been invited to join a team External Inbox ×

DrivenData <noreply@drivendata.org> Thu, Dec 15, 1:00 PM (22 hours ago)
to me

Hi there,

User **PVKNJX** has invited you to join the team **A_DIV_KLETECH** for the competition **DengAI: Predicting Disease Spread** — please follow this link to confirm or decline the invitation:

https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/team/confirm/4a3d1f1727243f91a67dfd564e59bced68c90ada525244388cefa33517733036/

Happy modeling!

- The DrivenData Team

unsubscribe

Figure 20: Team Member 3

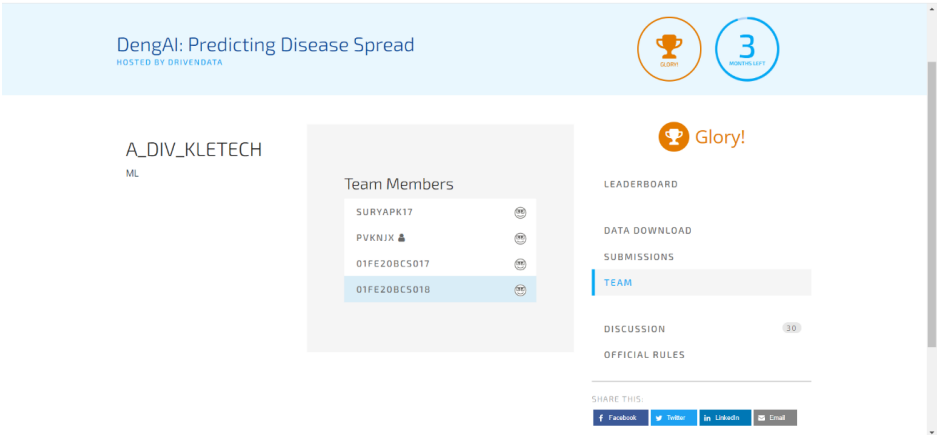


Figure 21: Total team members

References

[1] Sathler, Carlos. (2017). Predictive Modeling of Dengue Fever Epidemics: A Neural Network Approach..

[2] Albasheer, Osama Abdelmola, Amani Alomaish, Abeer Dallak, Alanood Darraj, Kawakeb Hamzi, Marwah Shawlan, Hanan Sumaily, Shareefa Gomairy, Marwah Hakami, Rana Mahfouz, Mohamed. (2021). Dengue paper PDF. Medical Science. 25

[3] Hasan S, Jamdar SF, Alalowi M, Al Ageel Al Beaiji SM. Dengue virus: A global human threat: Review of literature. J Int Soc Prev Community Dent. 2016 Jan-Feb;6(1):1-6. doi: 10.4103/2231-0762.175416. PMID: 27011925; PMCID: PMC4784057.

[4] P. Siriyasatien, S. Chadsuthi, K. Jampachaisri and K. Kesorn, "Dengue Epidemics Prediction: A Survey of the State-of-the-Art Based on Data Science Processes," in IEEE Access, vol. 6, pp. 53757-53795, 2018, doi: 10.1109/ACCESS.2018.2871241.

[5] Centers for Disease Control and Prevention. url: http://www.cdc.gov/

[6] Maria F. Vincenti-Gonzalez et al. "Spatial Analysis of Dengue Seroprevalence and Modeling of Trans mission Risk Factors in a Dengue Hyperendemic City of Venezuela". In: PLOS Neglected Tropical Diseases 11.1 (Jan. 2017), pp. 1–21. doi: 10.1371/journal.pntd.0005317. url: https://doi. org/10.1371/journal.pntd.0005317.

[7] U.S. Department of Commerce. url: https://www.commerce.gov/

[8] National Oceanic and Atmospheric Administration. url: http://www.noaa.gov/.

[9] Laurens van der Maaten and Geoffrey Hinton. "Visualizing Data using t-SNE". In: Journal of Machine Learning Research (2008). url: http://www.jmlr.org/papers/v9/vandermaaten08a.html.

[10]] Dengue Virus Net. [online] Denguevirusnet.com. Available at: http://www.denguevirusnet.com [Accessed 20 Jul. 2018].

[11] dengueforecasting.noaa.gov. (2018). Dengue Forecasting. [online] Available at: http://dengueforecasting.noaa.gov/ [Accessed 19 Jul. 2018].

[12] en.wikipedia.org. (2018). San Juan, Puerto Rico. [online] Available at: https://en.wikipedia.org/wiki/San-Juan,-Puerto-RicoGeography [Accessed 19 Jul. 2018].

[13] en.wikipedia.org. (2018). Iquitos. [online] Available at: https://en.wikipedia.org/wiki/IquitosGeography [Accessed 19 Jul. 2018].

[14] DengAI: Predicting Disease Spread - Benchmark - DrivenData Labs. [online] Available at: http://drivendata.co/blog/dengue-benchmark/ [Accessed 20 Jul. 2018].

[15] Random forest. [online] Available at: https://en.wikipedia.org/wiki/Random-forest [Accessed 20 Jul. 2018].

[16] How to Create an ARIMA Model for Time Series Forecasting with Python. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/ [Accessed 20 Jul. 2018].

[17] statsmodels.tsa.statespace.sarimax.SARIMAX — statsmodels 0.9.0 documentation. [online] Available at: http://www.statsmodels.org/dev/generated/statsmodels.tsa.statespace.sarimax.SARIMAX.html [Accessed 20 Jul. 2018].

Figure 22: Rank Obtained

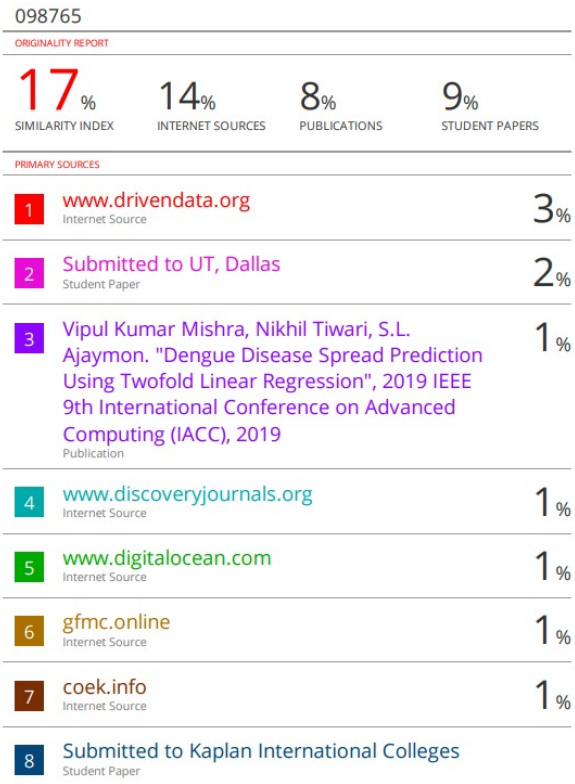


Figure 23: Plagarism Report