DengAI: Predicting Disease Spread

Professor:

Dr. Lawrence V. Fulton

Team Members:
Ratnam Dubey

DRIVEN DATA
USERNAME: - RATNAM

TABLE OF CONTENTS

Preface

1. Problem Discussion

Dengue fever is a mosquito-borne disease that occurs in tropical and sub-tropical parts of the world. In mild cases, symptoms are like 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 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:

Using environmental data collected by various U.S. Federal Government agencies—from the Centers for Disease Control and Prevention to the National Oceanic and Atmospheric Administration in the U.S. Department of Commerce—can you predict the number of dengue fever cases reported each week in San Juan, Puerto Rico and Iquitos, Peru?

This is an intermediate-level practice competition. Your task is to predict the number of dengue cases each week (in each location) based on environmental variables describing changes in temperature, precipitation, vegetation, and more.

An understanding of the relationship between climate and dengue dynamics can improve research initiatives and resource allocation to help fight life-threatening pandemics.

2. Literature

Understood the medical terminologies from the various discussions from Driven Data and several other websites also understood various Classification models and Bayesian Recognition Procedure (BPR) from the different sources. Some of the sources are rich in parameter tuning also helped to predict the Cases Correctly.

https://www.drivendata.org/competitions/44/dengai-predicting-disease-
http://deeplearning.net/tutorial/logreg.html
http://xgboost.readthedocs.io/en/latest/R-package/xgboostPresentation.html
http://wgrass.media.osaka-cu.ac.jp/gisideas10/viewpaper.php?id=342
http://www.statmethods.net/advstats/glm.html

Various techniques were used in the Dengue detection and Classification all the techniques which I have discovered in the Competition are provided above. Most of the techniques were based on the time series learning but it will surely take more time to complete to enhance the process and get the better result I have correlation between the data and trained the model based on the inputs which I get. I Extracted the features from the R Code which take some time to get complete whereas I have trained the model using R code.

3. Data Mining / Cleaning

3.1 Data Understanding

The datasets we used in our project came from an on-going Driven Data (*source:* https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/*)* competition. Data has been provided in below files: -

1) Train features
2) Sample Submission Files
3) Test Features
4) Train Labels

3.2 Data Preparation and Feature identification

We need to predict the total cases for the given cities. Information need to be Extracted from the train labels and features files. With the help of R Script, we have extracted the features from the all the training and test data. Data extracted from the dcm files have columns like

1. city
2. year
3. weekofyear
4. week_start_date
5. ndvi_ne
6. ndvi_nw
7. ndvi_se
8. ndvi_sw
9. precipitation_amt_mm
10. reanalysis_air_temp_k
11. reanalysis_avg_temp_k

12. reanalysis_dew_point_temp_k
13. reanalysis_max_air_temp_k
14. reanalysis_min_air_temp_k
15. reanalysis_precip_amt_kg_per_m2
16. reanalysis_relative_humidity_percent
17. reanalysis_sat_precip_amt_mm
18. reanalysis_specific_humidity_g_per_kg
19. reanalysis_tdtr_k
20. station_avg_temp_c
21. station_diur_temp_rng_c
22. station_max_temp_c
23. station_min_temp_c
24. station_precip_mm
25. total_cases

Variables are only in integers.

3.3 Missing Values

Considering the datasets this shows that the dataset is incomplete and there is need to clean it from empty entries'. Infect every column except State , Date and Date of year all the values are missing some are in greater quantity some in less replaced the missing values with the mean value to maintain the data integrity.

```
# count missing values (as percent)
apply(train, 2, function(x)
  round(100 * (length(which(is.na(x))))/length(x) , digits = 1)) %>%
  as.data.frame() %>%
  `names<-`('Percent of Missing Values')
```

```
##                                        Percent of Missing Values
## city                                                         0.0
## year                                                         0.0
## weekofyear                                                   0.0
## week_start_date                                              0.0
## ndvi_ne                                                     13.3
## ndvi_nw                                                      3.6
## ndvi_se                                                      1.5
## ndvi_sw                                                      1.5
## precipitation_amt_mm                                         0.9
## reanalysis_air_temp_k                                        0.7
## reanalysis_avg_temp_k                                        0.7
## reanalysis_dew_point_temp_k                                  0.7
## reanalysis_max_air_temp_k                                    0.7
## reanalysis_min_air_temp_k                                    0.7
## reanalysis_precip_amt_kg_per_m2                              0.7
## reanalysis_relative_humidity_percent                         0.7
```

```
## reanalysis_sat_precip_amt_mm                                 0.9
## reanalysis_specific_humidity_g_per_kg                        0.7
## reanalysis_tdtr_k                                            0.7
## station_avg_temp_c                                           3.0
## station_diur_temp_rng_c                                      3.0
## station_max_temp_c                                           1.4
## station_min_temp_c                                           1.0
## station_precip_mm                                            1.5
## total_cases                                                  0.0
```

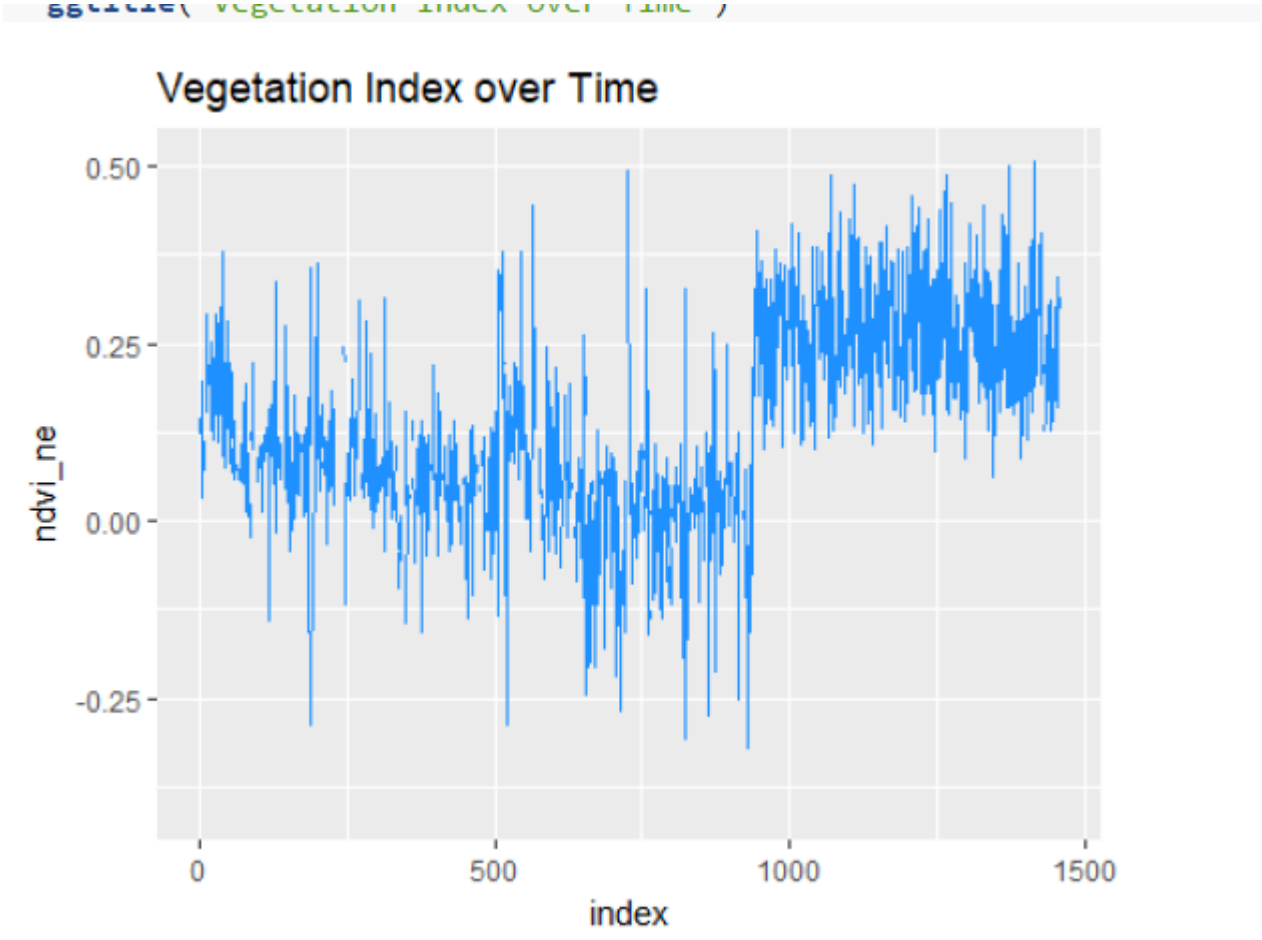
4. Data Visualization

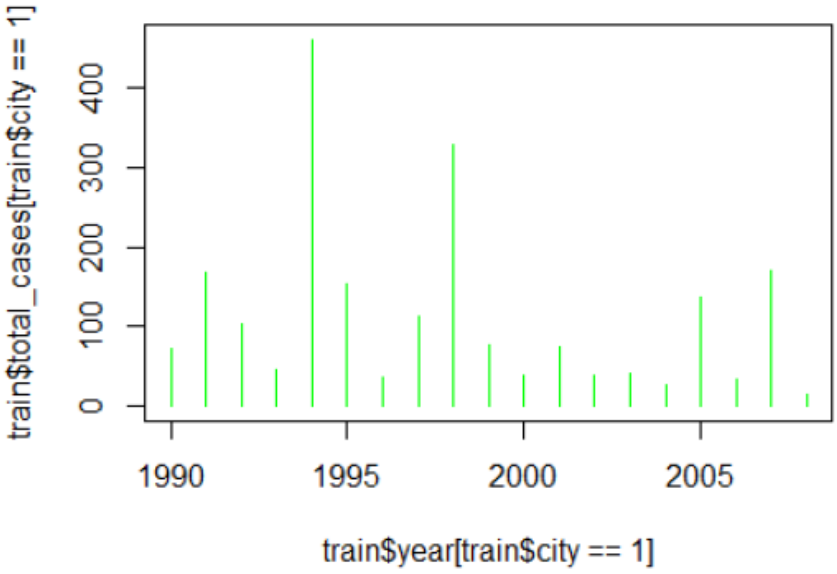
I used R to generate the models and for Data visualization. With the help of ggplot, plotly and scatterplot function generated several plots regarding the Data flow.

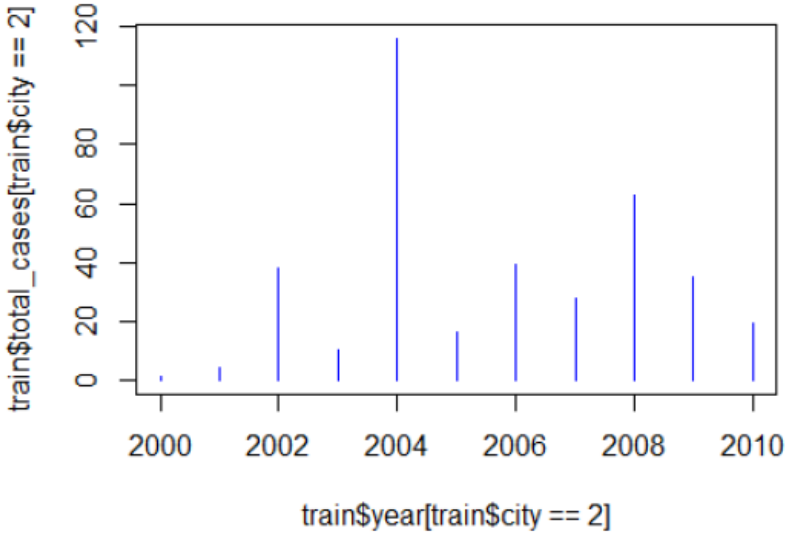
Sample Data visualization code for the Image generation using ggplot

```
# Plotting the Data
train %>%
  mutate(index = as.numeric(row.names(.))) %>%
  ggplot(aes(index, ndvi_ne)) +
  geom_line(colour = 'dodgerblue') +
  ggtitle("Vegetation Index over Time")
```

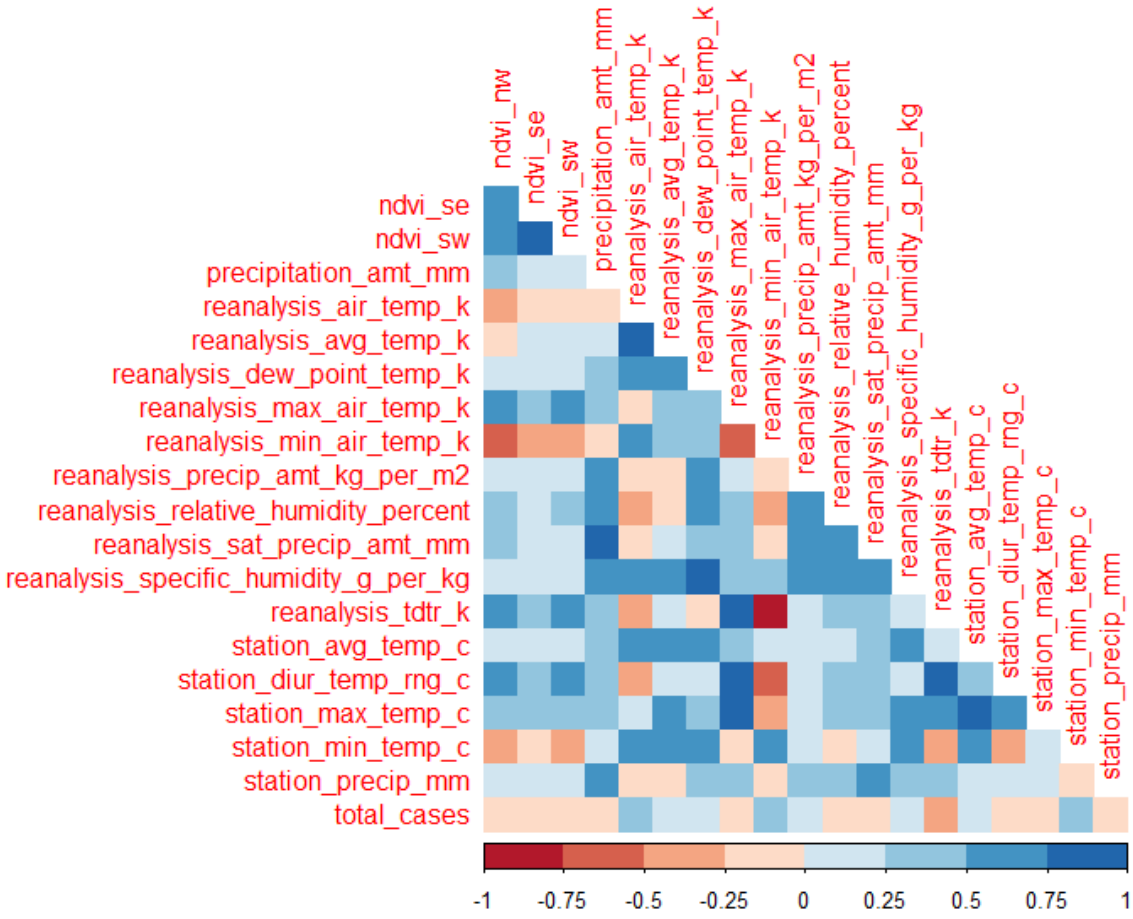
Some of the Generated images for the Data Visualization








```
plot(train$year[train$city==1],train$total_cases[train$city==1],type="h" ,
col="green")
```


ndvi_nw
ndvi_se
ndvi_sw
precipitation_amt_mm
reanalysis_air_temp_k
reanalysis_avg_temp_k
reanalysis_dew_point_temp_k
reanalysis_max_air_temp_k
reanalysis_min_air_temp_k
reanalysis_precip_amt_kg_per_m2
reanalysis_relative_humidity_percent
reanalysis_sat_precip_amt_mm
reanalysis_specific_humidity_g_per_kg
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
-1
-0.75
-0.5
-0.25
0
0.25
0.5
0.75
1

5. Predictive Techniques

I used R to generate of data and modelling the models. R seems to be an easy choice where we could do the analysis and Image Extraction in a quick time. To train the data I used R for the training the dataset. I have used plyr, dplyr, glm etc. package as the part data manipulation, file generation and modelling techniques.

Apart from glm algorithms using the python package, I also tried Logistic Regression algorithms. However, the best accuracy we got is with glm having tuned parameters and well-structured data. Board score of 26.52 which was better than the other models.

These output files are produced using 3 different models: -

- ✓ Logistic Regression in R
- ✓ GLM algorithm in R

5.1 Logistic Regression in R

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the 'multi_class' option is set to 'ovr', and uses the cross- entropy loss if the 'multi_class' option is set to 'multinomial'. (Currently the 'multinomial' option is supported only by the 'lbfgs', 'sag' and 'newton-cg' solvers.) This class implements regularized logistic regression using the 'liblinear' library, 'newton-cg', 'sag' and 'lbfgs' solvers. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied). The 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty.

5.2 GLM in R

glm is used to fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution.

Usage

```
glm(formula, family = gaussian, data, weights, subset,
    na.action, start = NULL, etastart, mustart, offset,
    control = list(...), model = TRUE, method = "glm.fit",
    x = FALSE, y = TRUE, contrasts = NULL, ...)

glm.fit(x, y, weights = rep(1, nobs),
        start = NULL, etastart = NULL, mustart = NULL,
        offset = rep(0, nobs), family = gaussian(),
        control = list(), intercept = TRUE)
```

6 Formulation / Libraries

We have used several libraries to analyze text Information. We have.

6.1 plyr

A set of tools that solves a common set of problems: you need to break a big problem down into manageable pieces, operate on each piece and then put all the pieces back together. For example, you might want to fit a model to each spatial location or time point in your study, summaries data by panels or collapse high-dimensional arrays to simpler summary statistics. The development of 'plyr' has been generously supported by 'Becton Dickinson'.

6.2 Dplyr

dplyr is a package for data manipulation, written and maintained by Hadley Wickham. It provides some great, easy-to-use functions that are very handy when performing exploratory data analysis and manipulation. Here, I will provide a basic overview of some of the most useful functions contained in the package.

6.3 ggplot

ggplot2 is a plotting system for R, based on the grammar of graphics, which tries to take the good parts of base and lattice graphics and none of the bad parts. It takes care of many of the fiddly details that make plotting a hassle (like drawing legends) as well as providing a powerful model of graphics that makes it easy to produce complex multi-layered graphics.

6.4 Plotly

An R package for creating interactive web graphics via the open source JavaScript graphing library plotly.js. plotly.js supports some chart types that ggplot2 doesn't (our cheat sheet provides a nice summary of the available chart types). You can create any of these charts via plotly().

7 Submission of models for scoring and kernel

We have Enhanced the model performance from *(Driven Data*) 27.895 to 26.739 by adding several features and by tuning the parameters.

8 Limitations

8.1 Process Time

Each model took a lot of time to process the execution which has certainly became a drawback because of which we couldn't make more hit and trails to the exiting model's.

8.2 Complexity

Most of the data was categorical which limits the usage of the models however we have changed into numeric values but that didn't show much improvement in the performance and model accuracy.

8.3 Data Driven Limitations

In **Data Driven** Prediction, we have limited upload of the submission file as due to which much more experiments are limited.

9 Learning

We have the more exposure to various algorithms and classifiers, we have learned to tune parameters. How to work with time series data. Taking the Date as the Column for the time series data and tuned the parameters regarding the same and generated the model for the same.

10 References

- ✓ XGB Boosting Machine- https://www.rdocumentation.org/packages/h2o/versions/3.10.0.8/topics/h2o.gbm
- ✓ Build A Big Data Random Forest Model- https://www.rdocumentation.org/packages/h2o/versions/3.10.0.8/topics/h2o.randomForest
- ✓ https://www.analyticsvidhya.com/blog/2016/05/h2o-data-table-build-models-large-data-sets/
- ✓ https://www.analyticsvidhya.com/blog/2016/05/h2o-data-table-build-models-large-data-sets/
- ✓ https://www.datarobot.com/blog/XGB-boosted-regression-trees/
- ✓ http://scikit-learn.org/stable/auto_examples/linear_model/plot_logistic_l1_l2_sparsity.html
- ✓ http://scikit-learn.org/stable/modules/preprocessing.html