DengAI: Predicting Disease Spread

Davide Lagano

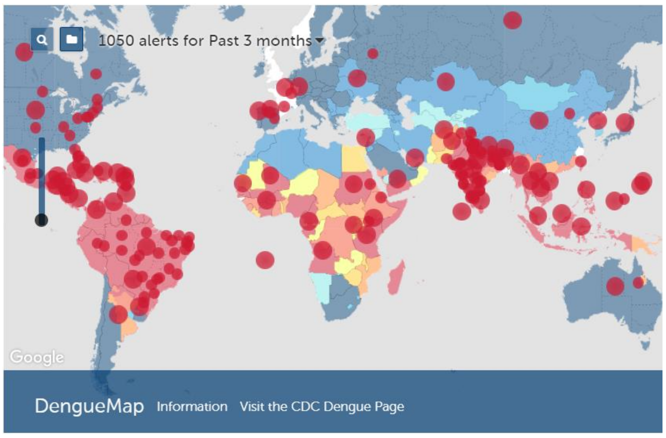
E-mail: d.lagano@campus.unimib.it

Abstract

For the project of DAMI_II, the problem "DengAI: Predicting Disease Spread" has been tried to be solved, provided by www.drivendata.org. This is an application-based project, that aims to define requirements and solve a particular application. The purpose of this paper is to predict the number of dengue fever cases reported each week in San Juan (Puerto Rico) and Iquitos (Peru) based on environmental variables describing changes in temperature, precipitation, vegetation and more. To solve the DengAI problem, different methods have been used, like: simple linear regression, XGboost, Negative binomial regression. Comparing the results is not possible since every method has different preprocessing. For this reason, the results are shown without comparing them.

1. Introduction

In this paper, a problem proposed in a competition of https://www.drivendata.org/, will be attempted to solve. As we can read in the competition: "Dengue fever is a mosquito-borne disease that occurs in tropical and sub-tropical parts of the world. In mild cases, symptoms are similar to the flu: fever, rash, and muscle and joint pain. In severe cases, dengue fever can cause severe bleeding, low blood pressure, and even death".
Because mosquitoes carry dengue, the transmission dynamics of dengue are related to climate variables such as temperature and precipitation. Although the relationship to climate is complicated, 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.



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

The dataset made available uses 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.

2. Purpose

The goal of the competition is to predict the number of dengue fever cases reported each week in San Juan (Puerto Rico) and Iquitos (Peru) based on environmental variables describing changes in temperature, precipitation, vegetation and more.

Performance is evaluated according to the mean absolute error.

3. Method

The challenge provides the dataset already split in three different parts: dengue_features_test, dengue_features_train, dengue_labels_train. After an accurate exploration of these datasets, it has been decided to do not consider the dataset 'dengue_features_test' because it is provided in order to be able to take part in the competition.

The project was started with importing the necessary library. The principals were pandas, numpy, matplotlib, seaborn, sklearn, scipy, xgboost.

3.1 Datasets

Afterwards the dataset was loaded. In this paper, the features are called ‘x’ and the labels ‘y’.

The features of the dataset are composed by 1456 rows and 24 columns:

City and date indicators:

- City, year, weekofyear, week_start_date.

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

- Pixel northeast of city centroid, pixel northwest of city centroid, pixel southeast of city centroid, pixel southwest of city centroid.

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

- Total precipitation.

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

- Mean air temperature, average air temperature, mean dew point temperature, maximum air temperature, minimum air temperature, total precipitation, mean relative humidity, total precipitation, mean specific humidity, diurnal temperature range.

NOAA's GHCN daily climate data weather station measurements:

- Average temperature, diurnal temperature range, maximum temperature, minimum temperature, total precipitation.

The labels dataset is composed by 1456 rows and 4 columns: city, year, weekofyear, total_cases (of dengue fever).

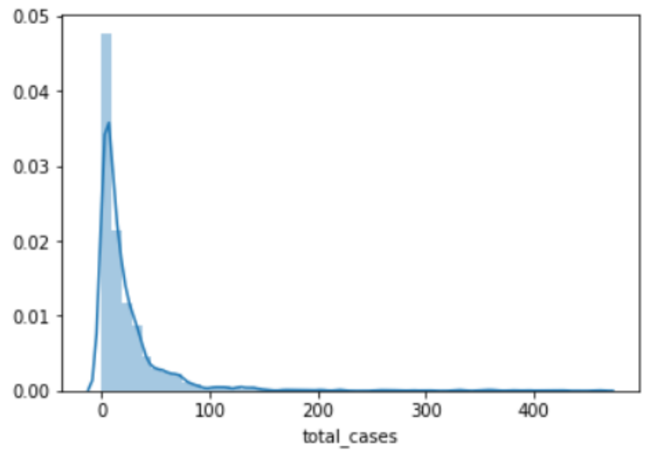
3.2 Explored datasets

After this first part, the datasets were explored.

As a first step, the type of columns were explored. Every column is 'float64’, except for the ‘city and date indicators'. It was not needed to convert these columns because they were useless for the classification phase.

Exploring the ‘total_cases’, it was seen that:

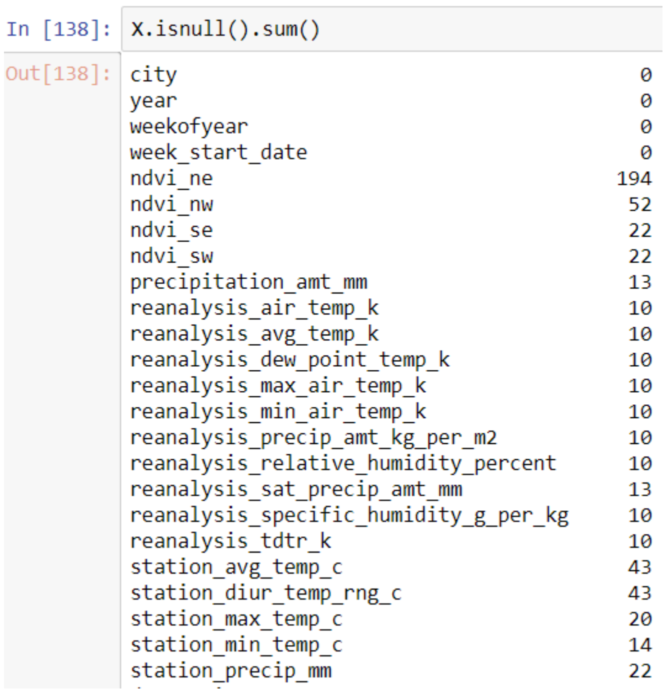
```
sns.distplot(y.total_cases)
plt.show()
```



As shown in the chart, the dataset has a highly skewed distribution. Combined with the fact that the labels are count variables, looking at the distribution it is clear that it is a good idea to try to implement a negative binomial regression.

3.3 Missing values

It is important to detect the missing value, and it is also important to solve the problem of the missing value properly. For this reason, missing values are explored in the dataset X:



```
In [138]: X.isnull().sum()

Out[138]: city                                    0
          year                                    0
          weekofyear                              0
          week_start_date                         0
          ndvi_ne                               194
          ndvi_nw                                52
          ndvi_se                                22
          ndvi_sw                                22
          precipitation_amt_mm                   13
          reanalysis_air_temp_k                  10
          reanalysis_avg_temp_k                  10
          reanalysis_dew_point_temp_k            10
          reanalysis_max_air_temp_k              10
          reanalysis_min_air_temp_k              10
          reanalysis_precip_amt_kg_per_m2        10
          reanalysis_relative_humidity_percent   10
          reanalysis_sat_precip_amt_mm           13
          reanalysis_specific_humidity_g_per_kg  10
          reanalysis_tdtr_k                      10
          station_avg_temp_c                     43
          station_diur_temp_rng_c                43
          station_max_temp_c                     20
          station_min_temp_c                     14
          station_precip_mm                      22
```

Fortunately, there are not many missing values. Regarding the dataset 'y', there is not any missing value.

It is decided to count the row in which the missing value is present, because if they are few, it is possible to delete them. The rows with missing values are 548.

4 Pre-processing

It is decided to use different preprocessing for each model. This is a controversial choice because it is not possible to compare the results of the different models, but the purpose of the challenge is to build the best model. After multiple attempts, it was evident that every model has better result with different preprocessing.

4.1 Simple linear regression pre-processing

The pre-processing of this model consists of converting the city column into a Boolean, since there are only two cities included. Extracting month from the date and removing the date info (city, year, weekofyear, week_start_date), because they are irrelevant for the classification. In this pre-processing, only the rows that contain NaN values are deleted.
Moreover, the StandardScaler function has been used in order to standardize the numerical features. Beside are extracted the month out of date and converts it to a one hot. The outliers have also been detected and removed. After this pre-processing part, the ‘y’ values are extracted and corresponding to the filtered ‘X’ values and used the ‘train_test_split’ function in order to obtain X_train, X_test, Y_train, Y_test.

4.2 XGBoost pre-processing

The pre-processing of this model has some parts that are the same as the previous one.
The 'missing value problem' was solved differently: the columns that had ~20% null values (ndvi_ne) were removed and ordered by feature importance. Then, the rest was filled using linear interpolation.
Also, converting the city column into a Boolean, since the problem has just two cities. Extracting month from the date, removing the dates info (city, year, weekofyear, week_start_date) because there were irrelevant for the classification.
Moreover, StandardScaler function has been used in order to standardize the numerical features. Beside are extracted the month out of date and converts it to a one hot.
Moreover, the data has been separated for the two cities in order to create a different model for each and split the dataset in train and test set for each city. The outliers were also detected the outliers and removed.

After this pre-processing part, ‘y’ values corresponding to the filtered ‘X’ values were detected and the ‘train_test_split’ function in order to obtain X_train, X_test, Y_train, Y_test has been used.

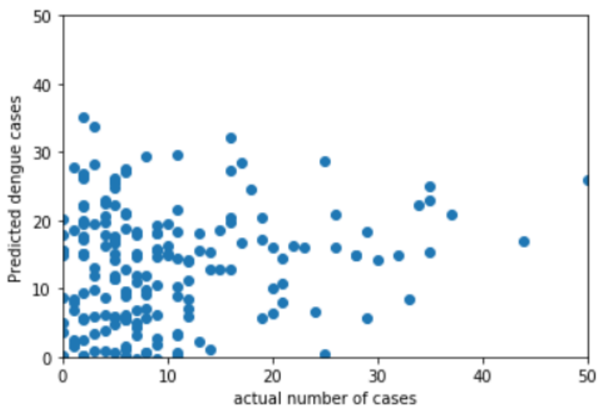
4.3 Negative binomial regression pre-processing

The pre-processing of this model has been slightly different confronted with XGBoost. Columns that had ~20% null values (ndvi_ne) were not removed.

5.1 Simple linear regression

Initially, in order to have an initial benchmark and compare the results of the model, a simple random model has been used.
Then is applied a baseline model using LinearRegression function.

Furthermore, it was decided to print the chart of predicted dengue cases and the actual number of cases. As we can see, it is not a good result.



In order to improve the result, a Feature Elimination using a base classifier has been applied, which is helpful to evaluate a subset of attributes. The RFE model was applied (which is basically a backward selection of the predictors) and after some attempt, 13 features were selected. Then it was split based on the newly selected features and train and test again.

The last step in order to obtain a better result was applying some regularization: the first regularization being L1, the second regularization being L2.

5.2 XGboost

Also, for this model, a simple random model has been used. The difference from the previous model is that in according to the pre-processing that has been used, 3 random models are needed, one for each country and one for both.

Thus, a generalized model is applied that it is a combination of accuracy scores on different cities trained on different models.

Using an 'XGBRegressor' model, there was an improvement, but it was not a significant gain.
Then, it was decided to tune the model with these parameters:

```
rf = XGBRegressor(n_estimators=55, learning_rate=0.01 ,n_jobs=-1, subsample=0.9, colsample_bytree=0.6,
                  colsample_bylevel=0.1, min_child_weight=5, reg_alpha=0.1)
```

After tuning an 'XGBRegressor' model, a good improvement was obtained.

As the final step, the cross-validation was used for both cities in order to evaluate the model. Result of SJ:

```
array([30.22416252, 49.28706257, 15.15194373, 32.01241734, 12.82045352])
```

result of IQ:

```
array([3.25735261, 7.91149119, 7.00273406, 5.89793904, 3.53914039])
```

As we can see from the arrays, there is a big variance from the two arrays.

5.3 Negative binomial regression

As a first step, the results of the separate cities SJ and IQ were examinated.

As in the XGboost, scores of the SJ and IQ cities were combined.

CV error = mean absolute error of predictions_sj and Y_sj_test. Used to estimate the test error better.
train error = mean absolute error of pred_train_sj and Y_sj_train.

6. Results

6.1 Simple linear regression

Simple random model performs a mean absolute error equal to 18.12.

LinearRegression function performs a mean absolute error equal to 12.63.

After the RFE model, the mean absolute error is 9.70.

L1, perform a mean absolute error of 8.20.

L2, perform a mean absolute error of 7.97.

6.2 XGboost

SJ random models perform a mean absolute error equal to 28.03

IQ random models perform a mean absolute error equal to 8.29

SJ+IQ random models perform a mean absolute error equal to 20.34

With 'XGBRegressor' model is obtained a mean absolute error of SJ+IQ equal to 18.27.

With boosted 'XGBRegressor' model is obtained a mean absolute error of SJ+IQ equal to 12.60.

6.3 Negative binomial regression

SJ results:
cv error: 22.61
train error: 25.48

IQ results:
cv error: 8.08
train error: 5.44

SJ + IQ
cv error: 17.42
train error: 18.32

7. Related works

https://towardsdatascience.com/using-keras-and-tensorflow-to-predict-dengue-fever-outbreaks-99392202bd5c

https://towardsdatascience.com/dengue-fever-and-how-to-predict-it-a32eab1dbb18

https://www.draper.com/news-releases/predicting-dengue-fever-outbreaks-machine-learning

https://new.hindawi.com/journals/cmmm/2019/7307803/

https://bmcinfectdis.biomedcentral.com/articles/10.1186/s12879-019-3874-x

https://www.gregcondit.com/projects/dengue-fever

8. Conclusion

For the first model, simple linear regression is better than a random model.
Moreover, recursive feature elimination improves the accuracy.
Furthermore, regularization helps to fix the overfitting.

Xgboost obtain a good mean absolute error, but applying the cross-validation it is evident that there is a big variance between the two arrays. As a future improvement of the paper, it could be interesting to indagate why in the cross-validation the array of the different cities are so different. Probably the improvement of the tuned model is due to the overfitting.

The results of the negative binomial regression are quite nice, it is impossible to compare these results with the others, but probably this is the model to start with to improve the paper.

References

www.drivendata.org