DengAI: Predicting Disease Spread

Alma, Cihan
Department of Computer Science
TOBB ETU, Ankara, 06510
calma@etu.edu.tr

Demirezen, Onur
Department of Computer Science
TOBB ETU, Ankara, 06510
odemirezen@etu.edu.tr

Karaduman, Yiğit
Department of Computer Science
TOBB ETU, Ankara, 06510
ykaraduman@etu.edu.tr

Şahin, Mehmet Arif
Department of Computer Science
TOBB ETU, Ankara, 06510
mehmetarifsahin@etu.edu.tr

Abstract

Dengue fever is a mosquito-borne disease and causes very serious health problems. Because it is carried by mosquitoes, the transmission dynamics of dengue are related to climate variables such as temperature and precipitation. Project goal is predict the number of dengue fever cases each week reported in San Juan, Puerto Rico and Iquitos, Peru under the scope of the *DengAI: Predicting Disease Spread* competition on *www.drivendata.org* website. In order to achieve this goal, some machine learning models were trained and tests were done using dataset arranged for the competition. This report includes the description of the dataset used for the prediction, the analysis and results of the tried methodologies, and finally conclusions reached about the project.

1 Brief description of the project

Project goal 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. To make this prediction, a model trained with dataset was created using machine learning approaches. What this project wants to achieve is that the error obtained when our model calculates the total case on the test data is as minimum as possible. Mean absolute error is used when calculating this error. Predictions of the models obtained from different methods were submitted in the competition and scores (mean absolute error values) are given. Throughout the project work, tried to improve this submission score value.

2 Dataset description

The dataset contains an climate, weather and environment values of two cities, San Juan and Iquitos, week by week as a feature. In dataset, San Juan has values for weeks between 1990 and 2008, and Iquitos has values for weeks between 2000 and 2010. Dengue fever disease is caused by mosquitoes and it is thought that there is a correlation between the spread of the disease and these variables. 20 distinct features were used and these features were collected from the following measurements: NOAA's GHCN city and date indicators, daily climate data weather station measurements, PERSIANN satellite precipitation measurements (0.25x0.25 degree scale), NOAA's NCEP Climate Forecast System Reanalysis measurements (0.5x0.5 degree scale), Satellite vegetation - Normalized difference vegetation index (NDVI) - NOAA's CDR Normalized Difference Vegetation Index (0.5x0.5 degree scale) measurements. For further information, you can check the problem description section in the official website.

Preprint. Under review.

3 Methodology

Different approaches were tried while the model was being trained. These approaches are linear regression, ridge regression, lasso regression, support vector regression, negative binomial regression, neural network and random forest. In this section, the consistency of the models created with these different approaches, the results of the prediction and the analysis are explained in detail.

3.1 Regression

Linear, ridge, lasso regression Studies have been done on whether dataset can be solved with classic regression methods or not. According to this, regression methods are implemented and trained with given dataset on Python. Because of the ridge and lasso restrict the coefficients against unnecessary growth, they performed better than the linear regression in some cases. We changed the α (shrinkage parameter) value on the ridge and lasso regression and looked at the variation of mean absolute error. The obtained error values and analysis are explained below. The average error values given after making the predictions on the test dataset separated from the given dataset. As expected before predictions, linear regression gave the error values that other methods gave when the shrinkage parameter was zero. As the α increases, the level of regularization increases, while lasso has made feature selection by making zero of some coefficients.

$$\text{Linear regression} \rightarrow \text{Mean absolute error: } 27.371$$

Table 1: Comparison of lasso and ridge regression for distinct shrinkage parameters

	Ridge regression		Lasso regression	
The shrinkage parameter	MAE	Number of features	MAE	Number of features
0	27.288	20	27.358	20
0.001	27.224	20	27.183	19
0.01	27.205	20	27.058	14
0.1	26.843	20	26.787	8
1	26.227	20	-	0
10	28.096	20	-	-
100	28.355	20	-	-

After changing the α, ridge and lasso shrink the coefficients and made regularization. Underfit observed when α value increased too much. After many tries on local training and test datasets, ridge regression gives the best results when the α is 1 and lasso regression gives the best results when the α value is 0.1. Therefore, we submitted the prediction results of this methods and take these scores:

$$\text{Ridge regression when } \alpha = 1 : 26.6875$$
$$\text{Lasso regression when } \alpha = 0.1 : 26.7572$$

Support vector regression Another regression algorithm we tried is Support Vector Regression which is a special version of SVM (Support Vector Machine) approach. We tried this algorithm because dataset has 20 features and SVR's computational complexity does not depend on the dimensionality of the input space.(1) We trained our SVR models with different parameters and make predictions. We tried different kernels in our model which are linear, polynomial and RBF (Radial Basis Function) kernel. Linear Kernel does not fit our dataset. Therefore, we did not continue to work on it. The test results we obtained from our dataset, submission scores and their analysis given below.
C->regularization parameter, a small C was encouraging a larger margin therefore it gave us a higher bias. The opposite situation occurred at very large C values.
Gamma-> Kernel coefficient. A small gamma will give us low bias and high variance while a large gamma will give us higher bias and low variance.

Table 2: Radial basis kernel function test results

γ/C	0.001	0.01	0.1	1	10
0.001	26.6226	26.5814	26.2607	25.9228	25.8473
0.01	26.5849	26.295	25.8113	25.6689	26.3859
0.1	26.6123	26.3173	25.7941	26.4905	29.9845
1	26.6346	26.6346	26.6792	28.3491	30.9605
10	26.638	26.638	26.7255	28.6054	31.2813

Table 3: Polynomial kernel test results

γ/C	0.001	0.01	0.1	1	10
0.001	26.6992	26.6878	26.6878	26.6878	26.6329
0.01	26.6878	26.4329	26.5797	26.5677	26.7049
0.1	26.566	26.7032	27.9348	33.186	42.0548
1	33.1852	42.0651	68.4888	96.9692	96.9485
10	99.312	96.9691	96.9691	99.3224	98.9691

Results After the test results, we observed best parameters to minimize the results and submit the predictions.

$$\text{Radial Basis Kernel Function } (\gamma = 0,01, C = 1) : 29.5361$$
$$\text{Polynomial Kernel } (\gamma = 0.01, C = 0.01, degree = 3) : 28.6731$$

3.2 Neural networks

Data preperation Data has been split considering *San Juan* and *Iquitos* are separate locations. All operations described in this section are common for data of two distinct cities unless otherwise specified. According to (2), when a person is infected with Dengue disease, it takes 4-5 days to show symptoms. Fever is the main symptom since it can easily be detected. Average infected person consults to medical authorities in approximately 5 days into the fever. With this information, we can say that in average a person will appear approximately 1 week after the infection in total case count. Therefore, train labels are shifted down by one row for features to more accurately represent total case count. Another way to express the situations, the week *W* of year *Y* corresponds to week *(W+1)* of year *Y* in the new dataset.
To fill missing values in dataset, last valid observation forward to next valid backfill is propogated.

Feature selection and normalization Features has been selected according to their correlation with target value *total_cases*. Correlation histogram can be seen in Fig. 1.

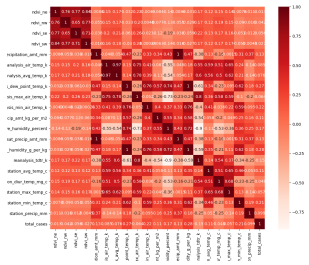

Figure 1: Correlation histogram of the data

We can see no significantly high correlation but there are features that has significantly low correlation with target. Therefore, a threshold of 0.07 has been determined. 13 features out of 20 that has higher correlation than specified $threshold$ are selected to train data.
To normalize the data, a min-max scaler function that fits all the data in range $[0, 1]$ is used.

Building the model According to Jeff Heaton in (3), a neural network containing one hidden layer can approximate any function that contains a continuous mapping from a finite space to another, and the number of hidden neurons should be $^2/_3$ of the size of the input layer, plus the size of the output layer. Since we have 13 features to build and train the model, and 1 target value; 13 input neurons, one hidden layer containing 10 neurons and 1 output neuron are generated as demonstrated in Fig. 2.

$RMSprop$ algorithm with a learning rate of $^1/_{10000}$ is used as the optimizer along with mean absolute error (specified in project description) as the loss function.

Fitting data to model While fitting the model, $epoch$ number is determined emprically. Model underfits below ≈ 200 $epochs$ and overfits above ≈ 1500 $epochs$ for *San Juan*. Numbers are below

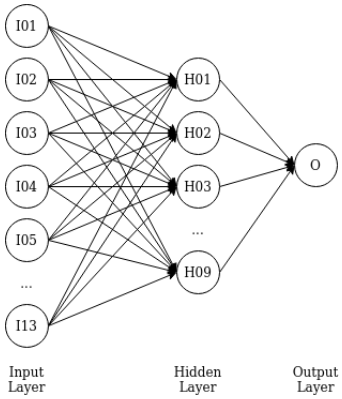

Figure 2: Graph representation of neural network

≈ 50 for underfit and above ≈ 750 for overfit in *Iquitos* data.
Best results are observed in 680 epochs for *San Juan* and 180 epochs for *Iquitos*.

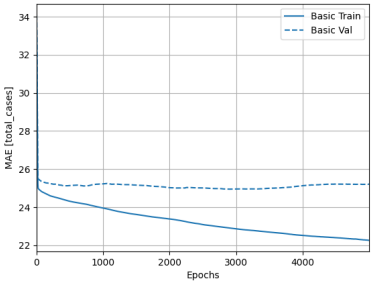

Figure 3: *San Juan* MAE change relative to epochs during training

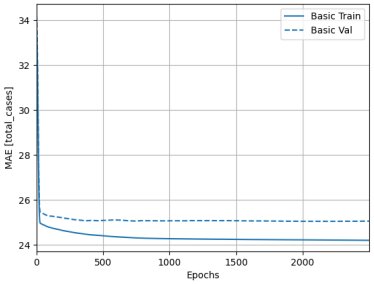

Figure 4: *Iquitos* MAE change relative to epochs during training

$Validation\ split$ is determined 0.33 since test data is $\approx 1/3$ of training data. Data is time sequenced, therefore $shuffling$ during training is disabled.

Results After a considerable number of trials, best score we got using neural network regression is 28.6034 in the form of mean absolute error.

3.3 Negative binomial regression

Motivation Negative binomial regression is basically for modeling count variables, usually for over-dispersed count outcome variables. Often, the variance is greater than the mean, a property called over-dispersion, and sometimes the variance is less than the mean, called under-dispersion. In such cases, one needs to use a regression model that will not make the equi-dispersion assumption which is $variance = mean$. The Negative Binomial (NB) regression model is one such model that does not make the $variance = mean$ assumption about the data. As it has mentioned before, our goal for the project is predicting the counts of total-cases of two cities, which is directly related with modeling count-based models.

Examples of negative binomial regression ***Example 1.*** School administrators study the attendance behavior of high school juniors at two schools. Predictors of the number of days of absence include the type of program in which the student is enrolled and a standardized test in math.(4)
Example 2. A health-related researcher is studying the number of hospital visits in past 12 months by senior citizens in a community based on the characteristics of the individuals and the types of health plans under which each one is covered.(4)

Essential variables for implementation Let us start with defining some variables first. **Y is the vector of total case counts seen on given timeframes in our data**. Thus $Y = [y_1, y_2, y_3, \ldots, y_n]$. **X is the matrix of predictors a.k.a. regressors**. **λ is the vector of event rates**. It contains n rates $[\lambda_0, \lambda_1, \lambda_2, \ldots, \lambda_n]$, corresponding to the n observed counts in the counts vector y. The rate λ_i for observation i is assumed to drive the actual observed count y_i in the counts vector y. In addition, λ vector is a deduced variable that we calculated by the our model during the training phase. We mentioned that NB models does not make the $variance = mean$ assumption about the data. Instead, it requires $variance = mean + \alpha * mean^p$. We used NB2 model so the p equals 2.
Finding the alpha: Cameron and Trivedi suggest a clever way to calculate alpha using a technique they call auxiliary OLS regression without a constant in their book called Regression Analysis of Count Data. (6)

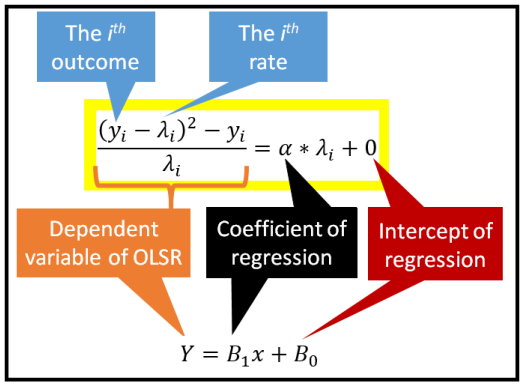

Figure 5: Process of finding the α value (5)

We can find the value of α, once we have fitted the auxiliary regression equation using the Ordinary Least Squares Regression technique on our data set of counts. To find λ_i, we fit the Poisson regression model to the our data set. In fact, doing so gives us the complete rate vector $\lambda = [\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_n]$ corresponding to all n observations in the data set.

Implementation Since, we mentioned all the ingredients that we need to implement NB2, we started the implementation by training Poisson Regression to find the alpha. We used the *Python statsmodels GLM class* for this training process. Here is the result:

```
CITY SJ POISSON
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:              total_case   No. Observations:                  936
Model:                             GLM   Df Residuals:                      916
Model Family:                  Poisson   Df Model:                           19
Link Function:                     log   Scale:                          1.0000
Method:                           IRLS   Log-Likelihood:                -16774.
Date:                 Thu, 23 Jul 2020   Deviance:                       29091.
Time:                         21:10:21   Pearson chi2:                 3.83e+04
No. Iterations:                      5
Covariance Type:             nonrobust
==============================================================================================
                                 coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------------------
Intercept                     71.8232     24.557      2.925      0.003      23.693     119.953
ndvi_ne                        0.0626      0.076      0.819      0.413      -0.087       0.212
ndvi_nw                        0.7537      0.086      8.742      0.000       0.585       0.923
ndvi_se                       10.5730      0.161    -65.768      0.000     -10.888     -10.258
ndvi_sw                        9.5752      0.167     57.274      0.000       9.248       9.903
precipitation_amt_mm          -0.0013   8.81e-05    -14.955      0.000      -0.001      -0.001
reanalysis_air_temp_k          2.1640      0.321      6.746      0.000       1.535       2.793
```

reanalysis_avg_temp_k	-1.5179	0.076	-19.919	0.000	-1.667	-1.369
reanalysis_dew_point_temp_k	-1.5042	0.297	-5.056	0.000	-2.087	-0.921
reanalysis_max_air_temp_k	0.4484	0.017	25.943	0.000	0.415	0.482
reanalysis_min_air_temp_k	0.0847	0.020	4.218	0.000	0.045	0.124
reanalysis_precip_amt_kg_per_m2	0.0013	0.000	6.673	0.000	0.001	0.002
reanalysis_relative_humidity_percent	0.1788	0.065	2.730	0.006	0.050	0.307
reanalysis_sat_precip_amt_mm	-0.0013	8.81e-05	-14.955	0.000	-0.001	-0.001
reanalysis_specific_humidity_g_per_kg	0.7089	0.080	8.859	0.000	0.552	0.866
reanalysis_tdtr_k	-0.4475	0.020	-22.192	0.000	-0.487	-0.408
station_avg_temp_c	-0.1863	0.019	-9.794	0.000	-0.224	-0.149
station_diur_temp_rng_c	0.0827	0.012	6.780	0.000	0.059	0.107
station_max_temp_c	0.0482	0.010	5.013	0.000	0.029	0.067
station_min_temp_c	0.0269	0.012	2.294	0.022	0.004	0.050
station_precip_mm	-0.0001	0.000	-0.409	0.682	-0.001	0.000

After we found the λ vector, added it to our dataframe as a new feature column. Next, we applied the auxiliary OLS regression. which we explained by it's formula above. As a result we found the alpha 1.163867. In the next step, we confirmed by t-value calculator that our alpha is statistically significant.
At the last step, we gave our alpha value as an input parameter to the NB2 model and made the fitting operations. Here is the result of our model's total case predictions in the city of San Juan.

CITY SJ

year	weekofyear	mean	mean_se	mean_ci_lower	mean_ci_upper
2008	18	15.312650	2.584377	10.999924	21.316260
	19	20.615503	2.360443	16.471483	25.802108
	20	12.171154	2.991657	7.518086	19.704084
	21	15.430028	3.826305	9.490468	25.086831
	22	25.790806	5.797811	16.600208	40.069717
...		...	...	...	...
2013	13	11.198419	1.775601	8.207139	15.279941
	14	19.217935	4.419062	12.245526	30.160324
	15	12.964109	2.216813	9.272364	18.125705
	16	9.894394	1.601062	7.205298	13.587090
	17	16.354831	2.457709	12.182389	21.956326

3.4 Random forest

One of the approaches we have tried was random forest(7) which looked promising due to a variety of reasons. One of the reasons was that random forest algorithm is very robust when it comes to dealing with complex problems such as this one. This characteristic was critically important due to the given dataset, a lot of other approaches are prone to get confused by noise, irregular targets, heterogeneous features etc. unlike random forest algorithm. Other major advantage of random forest was, because it is an ensemble method, it can handle a lot of features without a problem. This means that we can add such features that are slightly relevant, features that we cannot predict if it is relevant or not without the concern of confusing the model and decreasing its predictive power. Other advantage we can mention is that because no assumptions of linearity or normality has been made by the algorithm, it can handle correlation between features better than other approaches.
Advantages that are mentioned previously and variety of advantages comes implicitly with random forest algorithm makes it a strong candidate to predict with high accuracy. Our implementation of random forest on this problem started with manipulating the dataset for better results. First, we separated two cities in order to predict more accurately. We fill the missing data in our dataset with forward filling, meaning that we will propagate last valid observation forward until there is no missing

value in our dataset. Then, we get rid of the outliers in our training data. We reject the samples in our dataset that is smaller than minimum sample in provided test data and similarly; we reject the samples in our dataset that is greater than maximum sample in the provided test data. This helps that we do not fit our model to the outliers and helps with the accuracy. Figure 1 shows the difference between the accuracy of the model with and without the outlier detection technique is applied. Techniques that are used with outlier detection includes: Isolation Forest(8), Local Outlier Factor(9), and previously mentioned custom method that rejects the samples that are exceeding the bounds of the test data.

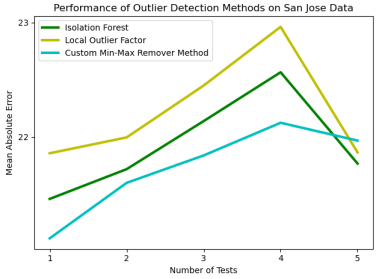

Figure 6: Outlier sample rejection

Then, we proceed to manipulate our data using *MinMaxScaler* to normalize the data. We selected the features that we believe are the ones that most relevant and took their minimum and maximum values and normalized our data between those two values. Random Forest algorithm gives us confidence to add more features without the fear of distracting our model.
This flexibility also allows us to add features that are few weeks lagging in order to help our model predict more accurately. We select the most relevant features that are mainly the ones that temperature values and humidity values; add one, two, three and four weeks prior data as features. Furthermore, we can group our data as sessions and add them as an extra feature. Because of the nature of this diseases, we expect to see reported cases after two or three weeks after conditions are met in order to spread the diseases. This added features helped our model to extract necessary information to predict the spread of the diseases. Figure 7 shows an example the importance between various feature selection decision.

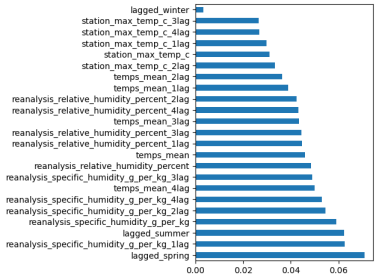

Figure 7: Feature importance chart

After the pre-processing is complete; we proceeded to tune our hyperparameters to accurately predict the values. In order to accomplish this, we have used RandomizedSearchCV from sklearn library that enabled us to give vague values to hyperparameter we empirically determined and find the close-to-best value for each. We tried *RandomizedSearchCV* with *5-fold cross validation*, 1000 iterations, Randomized Search, as its name would suggest, would randomly try hyperparameters; not all of them. After we get better idea which parameter should approximately with Randomized Search, we used Grid Search with Cross Validation a method tries all possible given hyperparameter value; and helps to know which one is the best. After various tries with different hyperparameter values, our effort with Random Forest yielded to Mean Absolute Error of 24.3918; achieved best results compared to other approaches.

4 Results

Table 3: Best submission scores of different methods

Methodology	Best submission score (MAE)
Random forest	24.3918
Ridge regression	26.6875
Lasso regression	26.7572
Negative binomial regression	26.9087
Neural network	28.6034
Support vector regression	28.6731

5 Conclusion

We created and tested models with non-complex approaches such as lasso and ridge regression. When the shrinkage parameter was selected appropriately, regularization could be done properly. Results were not considered bad according to methods other than random forest. Support vector regressor performed poorly. One of the reasons for that the SVG may not be suitable for large datasets with lots of features. Therefore, it may not have been able to fit properly. In addition to this, it can easily distracted with the noise in the dataset. It may be thought that it does not work well for such reasons. When it comes to neural networks, things become more complex. Since neural networks are very powerful tools, they easily overfit when not used properly. Neural networks have lots of parameters that requires time and experience to optimize. Therefore, we could not be much successful with them. Before NB2 implementation, we used a different regression to find alpha(OLS). In addition, lambda variables deduced in the training phase. These variables are given to NB2 regression directly. Hence, finding the accurate values for these factors plays a critic role to find appropriate prediction results by NB2. Therefore, this situation increase the accuracy risk in the NB2 regression. Random Forest was a strong candidate due to a variety of reasons. We concluded that random forest algorithm performs well because it is very robust. This was very important because of the type of the problem we are dealing with. Random forest is not prone to get confused by the noise, irregular targets, heterogeneous features etc. Because of the logic behind random forest, we can add features that are not very relevant without the concern of confusing the model and decreasing its predictive power. This feature of random forest led to the best results we have achieved.

References

[1] Awad, M. & Khanna R. (2015). *Support Vector Regression. In: Efficient Learning Machines*. Apress, Berkeley, CA: Chapter 4.

[2] Halstead, S.B. (2007). *Dengue*. Lancet 370, 1644–1652.

[3] Heaton, J. (2017). *The Number of Hidden Layers*.
Retrieved from: *https://www.heatonresearch.com/2017/06/01/hidden-layers.html*.

[4] *Negative Binomial Regression | Stata Data Analysis Examples*
Retrieved from: *https://stats.idre.ucla.edu/stata/dae/negative-binomial-regression/*

[5] *Negative Binomial Regression: A Step by Step Guide*
Retrieved from: *https://stats.idre.ucla.edu/stata/dae/negative-binomial-regression/*

[6] Cameron, A. & Trivedi, P. (1999). *Regression analysis of count data.* 2nd ed. 10.1017/CBO9780511814365.

[7] Breiman, L. (2001). *Random Forests*. Machine Learning. 45. 5-32. 10.1023/A:1010950718922.

[8] Liu, F. T. & Ting, K. M. & Zhou Z. (2008) *Isolation Forest*. 2008 Eighth IEEE International Conference on Data Mining. Pisa. pp. 413-422. doi: 10.1109/ICDM.2008.17.

[9] Breunig, Markus & Kriegel, Hans-Peter & Ng, Raymond & Sander, Joerg. (2000). *LOF: Identifying Density-Based Local Outliers*. ACM Sigmod Record. 29. 93-104. 10.1145/342009.335388.