DengAI: Predicting Dengue case numbers using climate data

*I have tried and tested many different models and combinations of hyper parameters, feature subsets, feature extractions, and variable scaling approaches.*

**LinearRegression:**

The first approach tried was a simple Linear Regression to get the ball rolling. This notebook also processes the data into the form that is used in later notebooks. Starting off, San Juan seemed to follow a more predictable annual pattern than Iquitos – but averaging out the years available in our dataset, we see a pattern emerge.

A first, though naïve approach, was to try modeling the case numbers off of the monthly overall average case numbers for each city. This involves creating dummy variables for each month, which is utilized in later notebooks. With the month features created as binary features, a Linear Regression is fit and the results are not accurate as predicted.

The cyclical pattern seen in both prediction lines is for each year in that particular test set, and we see some contrasting results for each city, which will become a trend throughout most of the modeling. The San Juan Linear Regression model seems to be over predicting for almost all of the weeks in the test set except for the 2 outbreak periods where it is still under predicting. This may be due to the fact that the training set it was fit on had many more periods of outbreaks and higher weekly case numbers than the test set displays, so it learned to predict higher case numbers as a result. Iquitos on the other hand, is under predicting for most of the test set. And this seems to be a result from the opposite condition that San Juan experiences, the training set had one outbreak period where the case numbers of the weeks exceeded the max case numbers observed in the testing set – the testing set had higher “outbreak” periods than the what the model was fit on with the training data and thusly learned to predict lower case numbers.

In order to use any of the other features made available, the missing values will need to be handled first. Simple methods of imputation, such as forward fill since this is a time series, would be okay for most of the features but some have large gaps of consecutive weeks of missing values and forward filling or a simple linear fill method of that nature would throw the models off due to not being able to correlate the different values for said feature with differences in case numbers.

Though it may be unnecessary, I’ve decided to convert the Kelvin unit temperature features to Celsius to align with the units of the other temperature features available. Once the different temperature feature units are aligned, there are still missing values in the dataset, and will need to be imputed. Simple methods like forward fill, or using the mean or median values of the feature produce flat and basic feature values relative to the variability of that feature. This isn’t a big deal for features with only 2 or 3 missing values, but are not ideal methods for features that have 20 to 30 missing values. I wanted to preserve some variability in the independent variables so models could learn on different values rather than fitting on flat values. At first, K-Nearest Neighbors imputer was used because our values displayed seasonality, generally reappearing a pattern annually, meaning if one year was missing a value for station minimal temperature, it’s likely there are other years where this value isn’t missing that the imputer could learn from. This approach was time consuming, which is why I turned to Bayesian Ridge imputation. There was a minimal difference between the imputed values between the two approaches but the time needed to perform Bayesian Ridge was far less than what was needed for KNN Imputation. With the difference in predicted values being so slight, I felt the impact of these differences was insignificant and preferred the faster results Bayesian Ridge provided.

With the data imputed, the independent variables were ready to be fit to a model, and the first model tried was again the Linear Regression model. I didn’t expect these predictions to be very accurate and that is seen in the scatter plots of predicted cases vs actual case numbers. Both models predict lower case numbers compared to the case numbers seen in the outbreak periods. And the line plot for San Juan looks almost identical to predicting with just the month variables. This may be due to the stronger annual trend seen in San Juan than the weaker annual trend of Iquitos. The linear prediction for Iquitos does still present a slight seasonality in the two years of test data in this case, but overall, the prediction values are higher, and interestingly there is a shock in case number predictions for 2010. Strangely, this doesn’t seem to align with an outbreak period in the actual case numbers of Iquitos. Not surprisingly, the Linear Regression model doesn’t seem to be the most appropriate approach for this circumstance.

**ARIMA Models**:

With this being a time series, a model of the ARIMA family may be a more appropriate approach. With ARIMA models, the time element of the dataset is very critical, and upon closer inspection of the data, the weekly recorded features and case numbers all start on the first of the year each year, and end on the 24th of December each year. This makes comparing the values in these weeks easier, but doesn’t provide a frequency in the sampling that ARIMA models can interpret since each sample for a given day and month would have a different weekday for consequent years and doesn’t provide a consistent frequency of samples. One way to correct this is to try and align all the weekly samples to the same weekday and to decide which portion of the reported cases for that week would end up falling into the subsequent week values, but I’ve decided to roll the case numbers into monthly samples instead. This may still leave some quantity of case numbers that would overlap with previous or succeeding months but the overlap would only result from the first and last weeks of the month rather than a small overlap of each week across the duration of the dataset.

With the data prepared for the ARIMA models, the first step would be to get the AR and MA values of the time series. Another important step is to examine the decomposition of the time series, breaking it down into the trend, seasonality, and residuals. The ideal dataset that is fed into these models is one that is stationary as many models need stationary time series to make accurate predictions. There are a few different kinds of stationary, though the one that is usually targeted is the second order, or weak stationarity, whereby the time series has constant mean, variance, and covariance. A few tests are available to test for stationarity, the two featured are the Augmented Dickey Fuller (ADF) unit root test and the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) trend stationarity test.

The ADF test proposes two hypotheses:

* **H\_0**: The series has a unit root.
* **H\_A**: The series has no unit root.

The KPSS test proposes:

* **H\_0**: The process is trend stationary
* **H\_A**: The time series has a unit root.

Based on this testing, both cities have a small p-value for the ADF unit root test, suggesting there is strong evidence to reject the null hypothesis. The KPSS test, another test for unit root in the time series often used to test for stationarity, suggests there isn’t strong enough evidence to reject the null hypothesis of stationarity. Statsmodels.org provides some suggestions on different possible outcomes of these two tests and suggests that in our case of stationary ADF and non-stationary KPSS, we can apply differencing to make the time series stationary, though this likely won’t help to make the time series strict stationary due to the outbreak periods present. I have tried many different differencing terms, first and second order differencing, up to 12 periods (months) of differencing, and detrending, but the time series wasn’t able to make the series stationary. This is why the function *getkpss* uses the regression = ‘ct’ parameter, to test whether the time series is stationary about a trend rather than about a constant, but for both time series, we get large p-values against the null hypothesis regardless of the value used for the regression parameter.

I’ve chosen some features to use to model the time series as trying to model or find the MA, AR, and integration elements for the trend order and the seasonality order for the SARIMAX model since trying to use all of the features available took far too long to run for each city – sometimes in the order of a few hours. At first, the tsfresh package was used to extract features and statistical methods of the time series, which itself took far too long to be practical and blew up the feature space to over 3500 features. Utilizing the fresh (FeatuRe Extraction based on Scalable Hypothesis tests) method found in the feature\_selection.relevance subclass, it eliminated almost all of the different features and aspects of the time series extracted anyways. This may be a good library to use for different time series analysis as some features like the variance of time may be more important to understanding the outcome, but when trying to fit the exogenous variables, my system was constantly running out of memory, and when using the selected features in city\_relevant, the time to extract and reduce the extracted features was over 3 hours. Due to this, I couldn’t utilize the library to its full potential due to my system limitations.