# Capstone Project Report

## Definition

### Project Overview

As a resident of the United Kingdom it is easy to forget about the existence of serious infectious diseases which have been largely eliminated here. However they are still a cause of great hardship and economic drag in many places and with changing climate patterns over the coming decades these diseases are likely to spread. One such disease is Dengue Fever which is common in Latin America and South East Asia, infecting up to 400 million people each year. Dengue fever is mosquito borne and therefore liable to spread as climate change increases the habitable range of these insects. Information about dengue fever and its epidemiology can by found on the Centre for Disease Control website at:

https://www.cdc.gov/dengue/index.html

Early warning systems for dengue outbreaks based on monitoring internet search terms have been tried in several places including China, see:

http://journals.plos.org/plosntds/article?id=10.1371%2Fjournal.pntd.0005354

Statistical / machine learning approaches have also been developed, over both short and medium timescales, for example

https://ij-healthgeographics.biomedcentral.com/articles/10.1186/1476-072X-14-9

http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0152688

This project is based on a dataset published as part of a machine learning competition on the Driven Data website. See

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

The dataset consists of environmental measurements, such as temperature and humidity, together with reported dengue fever cases, on a weekly basis over several years. These measurements relate to two cities that are geographically quite different, so represent two different versions of the same problem.

### Problem Statement

Infectious diseases like dengue fever are by nature episodic and it is usually not cost effective to be constantly on standby in case of an outbreak. The problem is to predict as accurately as possible the number of dengue fever cases expected, based on currently collected environmental data, so that resources can be more efficiently and effectively allocated. This is a multivariate regression problem involving time series data.

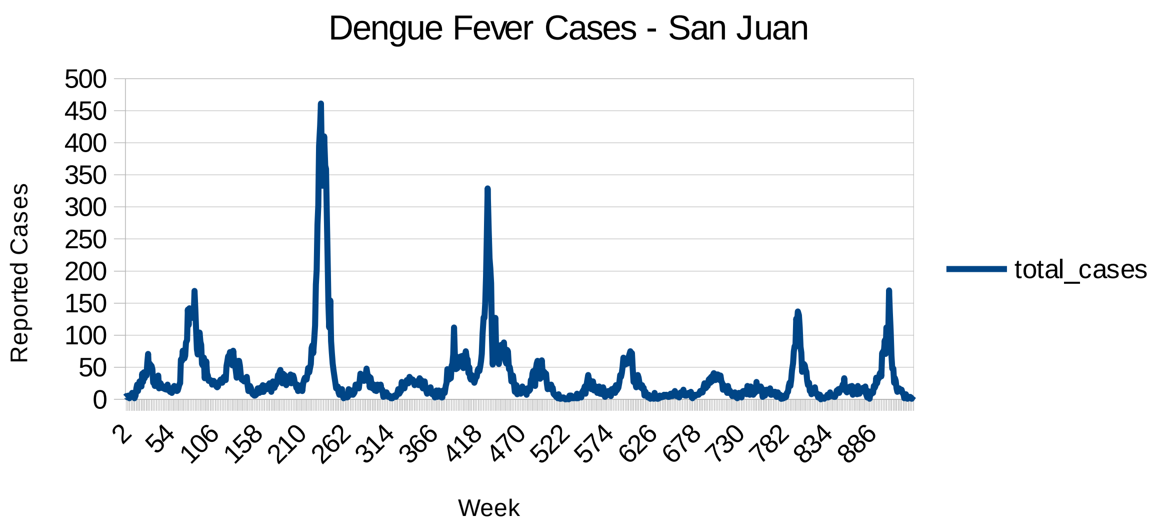
### Metrics

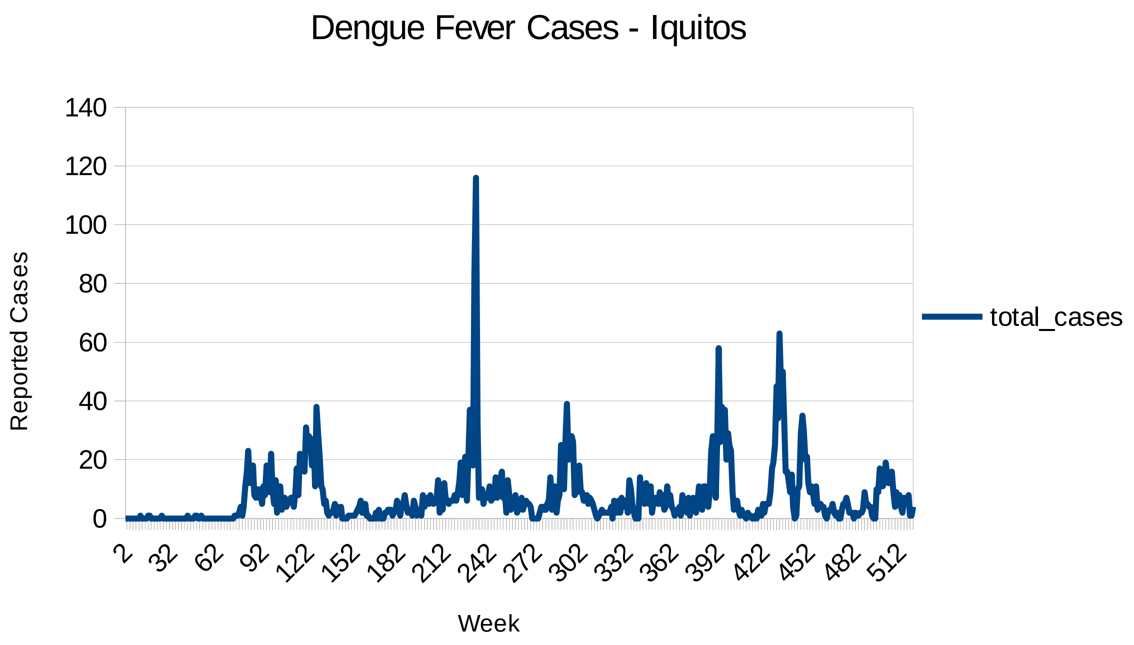
The competition on which this project is based stipulates the Mean Absolute Error as the metric to be used. However I believe that it would be better to use a metric which emphasises larger errors over small ones. I believe that a small error in predicting the number of dengue fever cases can probably be largely ignored since it would not impact the allocation of resources. So although I will be trying to optimise the Mean Absolute Error, as per the competition rules, I will also calculate the $R^2$ score. For any given test set the $R^2$ score reflects the mean square error, so this emphasises large errors over smaller ones. I prefer to use the $R^2$ score rather than the mean square error as the former is more easily compared with other machine learning situations.

## Analysis

### Data Exploration

The first step in this task is to look over the datasets to get a feel for the nature of the problem. The data is in two CSV files, one for the feature data and one for the labels data. The feature data consists of weekly measurements of 20 environmental parameters, taken at one of two city locations. The label data contains weekly reported dengue fever cases, again reported at one of two cities. The data ranges from mid 1990 to mid 2008 for San Juan, and from mid 2000 to mid 2010 for Iquitos. As these two locations are quite different I am going to treat them separately, at least to begin with. A plot of the number of reported dengue fever cases over time is shown below.



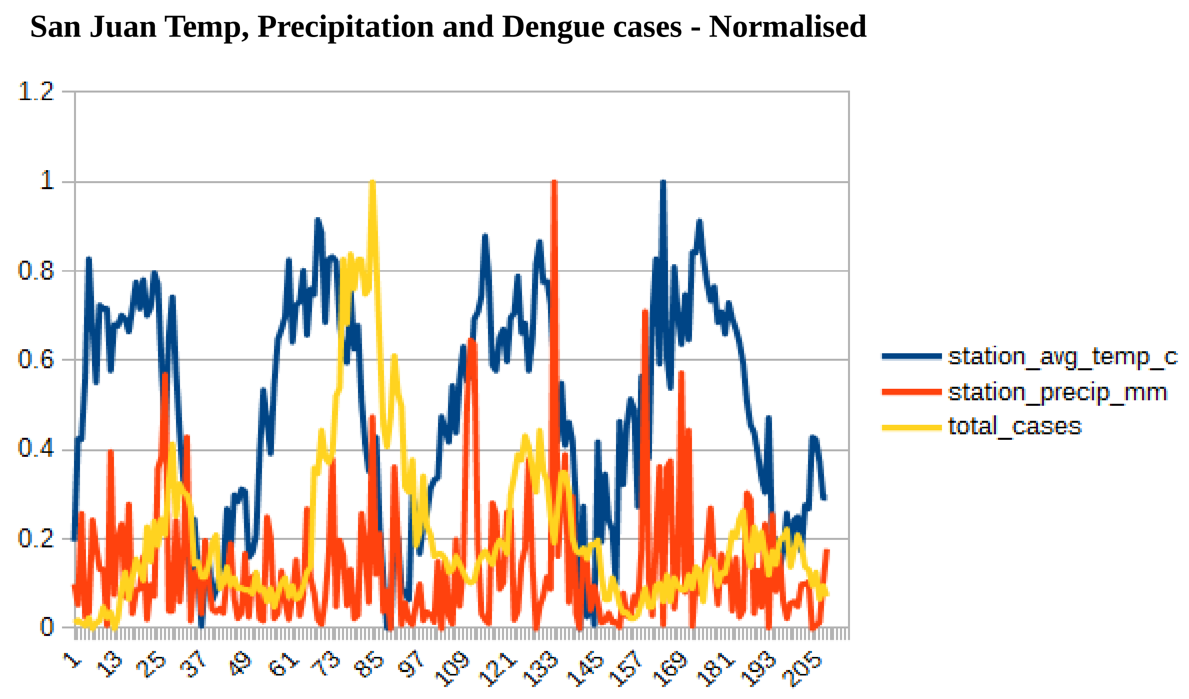


The data from San Juan shows quite clear annual cycles, but with large fluctuations from year to year. The Iquitos data looks noisier, in part because there are fewer cases overall, and the annual cycles are less clear (at least to the naked eye). Also, the first 70 weeks or so of the Iquitos data look anomalously low, perhaps because the reporting system for dengue fever cases was not fully set up at this point. Looking at a few simple descriptive statistics:

| **CITY / STAT** | Mean | Std Dev | Min | Max | Kurtosis | Mean Abs Dev |
|---|---|---|---|---|---|---|
| San Juan | 34.18 | 51.38 | 0 | 461 | 25.37 | 28.35 |
| Iquitos | 7.57 | 10.77 | 0 | 116 | 26.74 | 6.68 |

Although the reported cases data for the two cities look somewhat different to the naked eye, their statistics are quite similar, which is perhaps reassuring. The large kurtosis shows the nature of the data, with much of the variance due to infrequent outbreaks caused by sudden population explosions in mosquito numbers. This will make modelling the data a real challenge. The label data is complete, with no missing values which is good.

The features that I will be trying to predict the number of fever cases from are weekly environmental measurements, such as temperature, humidity etc. There are 20 separate measurements, but some have names like “reanalysis_avg_temp_k” which suggest that they contain essentially the same data as other features (i.e. “station_avg_temp_c”). However a brief look at these show that the re-analysis is not just a change of temperature scale. I am not sure what the exact relationship is between the original features and the “re-analysis” features. Also, some features such as “station_max_temp_c”, “station_min_temp_c” and “station_avg_temp_c” are likely to be highly correlated although still carrying potentially important separate information. Mosquitoes need warmth and water to breed successfully so I would expect the average temperature and the precipitation amount to be important features. A plot of these two features (from the San Juan data) together with reported dengue fever cases is shown below. I have scaled all the features to be in the range 0-1 so that the detail of each is clearly visible.



From this plot we can see:

1) The mean temperature shows strong annual cycles, as you would expect, with considerable noise.
2) The precipitation data is very noisy.
3) Peaks in reported cases lag the peaks in average temperature by several weeks.

We would expect peaks in fever cases to lag behind the conditions needed for mosquitoes since the mosquito population will take time to build up. The typical life cycle for a mosquito is about 2 weeks, and several such cycles may be needed to build up a large population. Also, the fever itself takes up to 6 days to present symptoms.
Since the early part of the mosquito life cycle depends on water, this suggests that the amount of precipitation will be important. So the variability of precipitation will likely make this a difficult prediction.

To have a chance at predicting dengue fever cases we will need to include features from several weeks before the prediction data. If we include all 20 features for 8 weeks this will be 160 features. This may lead to over-fitting problems, particularly as the data is quite noisy, so I will need to keep a close eye on this.

All of the features have some missing data points. Because I intend to use several weeks of past feature measurements for each prediction, these missing values will have a much greater impact. In the case of the San Juan data these missing data are mostly isolated weeks for which there are no measurements. The exceptions are for the satellite measurements of vegetation cover which have a period of 15 weeks with no data during the 4th year. If I intend to use any of these features I will need to remove these weeks and any following ones that would be affected (depending on how long a data history I use in the predictions). For the other missing data that is just for short periods I think I will interpolate the values.

## Algorithms and Techniques

The features in the dataset are quite heterogeneous, and as mentioned before I may end up with quite a few features. I am also aware that my laptop does not have a great deal of computational power. So for my initial modelling I intend to use some form of decision tree regressor, perhaps together with an ensemble technique such a bagging. Decision trees are scale invariant, not too computationally intensive, and tend to be good at ignoring irrelevant feature inputs. Due to the nature of population dynamics I would expect the model to need to be quite non-linear which should also be handled well by a decision tree. A possible downside will be the limited scope for controlling over-fitting in regression trees, although the use of an ensemble method should help with this.

## Benchmark

The competition on which this project is based includes a benchmark which can be seen here.
http://blog.drivendata.org/2016/12/23/dengue-benchmark/

This benchmark model first fills in missing data by simply duplicating prior measurements and then selects the 4 features that are most strongly correlated with the observed number of dengue fever cases. The data is split into a separate training and validation set for each city. A negative binomial regressor is then fitted to each training set for values of the ancillary parameter alpha varying from 10e-8 to 10e-3. The negative binomial distribution describes the number of successes in a series of trials that terminates after a predetermined number of failures. The alpha ancillary parameter is the inverse of the number of failures before the trial series terminates. The resulting models are used to make predictions on the validation sets and the best value of alpha is selected based on the mean absolute error of the predictions. The mean absolute error of these models is 22.08 for San Juan and 6.47 for Iquitos.

## Methodology

### Data Preprocessing

As mentioned, there are some missing data points in the feature set. For the San Juan data the “ndvi_ne” feature is particularly bad. A quick look at correlations between the vegetation features (ndvi_ne/nw/se/sw) shows that these features are somewhat correlated but not very strongly. However I decided to drop the ndvi_ne feature. This feature looks quite noisy to me (for instance, vegetation cover frequently changes by a factor of 2 over a one week period. That seems unlikely to me), and the data even goes negative on occasion.

Having dropped the ndvi_ne feature, I then used the interpolate function on the other features with a limit of 1 measurement in both directions, so this interpolates up to 2 missing values. I have to admit that I am not confident that this is the right way to go. Because I plan to use 8 or possibly more weeks worth of past measurements in my model, each missing data value will affect multiple weeks of prediction, so if I base the model only on complete feature sets, without interpolation, I would loose quite a few data points. On the other hand the validity of interpolating features like min and max temperature is not clear to me. I think that this is a decision that I may want to re-visit, but for now I am going to interpolate up to 2 missing values.

Next I create the time shifted versions of each feature and add them to the features data-frame. Thus for each X, I create X_1_weeks_ago, X_2_weeks_ago, etc. so that the model can be based on historical measurements as well as the current ones. Initially I tried adding 12 weeks of historical data, giving 247 features! This process introduces missing data at the beginning of each time shifted feature, and also propagates any missing data that was not filled in by the interpolation step. So after this I looked for rows with missing data and removed them, also removing the corresponding entry in the label data. For the San Juan data this reduced the number of data points from 936 to 886

### Implementation

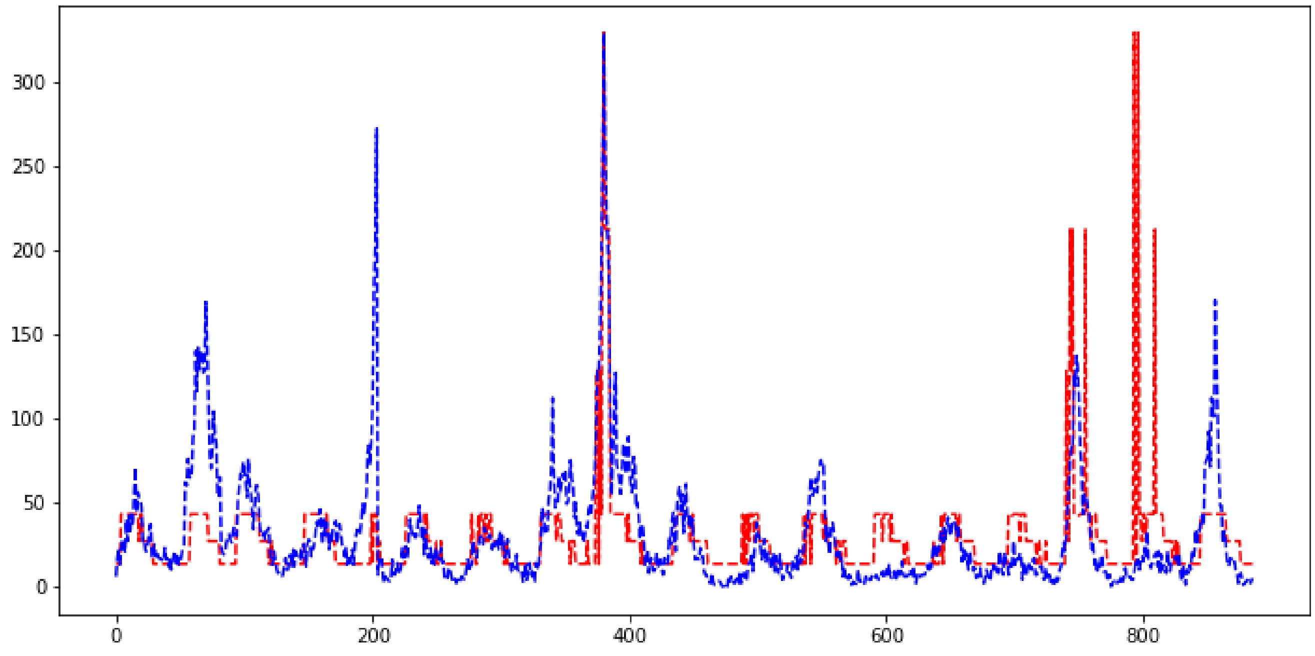
The model to be tried is a Decision Tree Regressor within an Adaboost ensemble method. The decision tree is not too computationally intensive, handles non linear models well and is good at ignoring irrelevant feature data. As it is scale invariant I have not normalised the feature data.

The Adaboost meta-estimator does not provide a mechanism to pass parameters to its underlying weak learner, so the parameters of the Decision Tree are fixed once we start using Adaboost. So I started by doing a parameter grid search using the basic Decision Tree Regressor. For cross validation I use a simple Kfold scheme where I set the number of folds to roughly split the data into 2 year chunks, without shuffling. Cross validation is an area where I initially made a big mistake. I initially created my own cross validator that split the data into sets containing every Nth data point. This was to ensure that during training the estimator looked at samples from right across the data set including all the peaks and all the troughs, while ensuring that the validation set also included all of these. However, because this is time series data, sequential points are highly correlated and my cross validation scheme allowed a lot of information to leak from the validation set into the training set (in fact the validation and training sets were almost the same, apart from the number of sample points!). This led to serious over-fitting which I did not realise until I submitted to the competition website and got a terrible result!

The grid search covered the max_depth parameter for values [3,4,6,10] and min_samples_split parameter for values [2,5,10,20]. I was somewhat surprised to find the best parameters were max_depth =3 and min_samples_split=2. Such a small tree depth seems very limiting, but I think that larger depths must be leading to over-fitting which impacts the cross validation scores. Using these parameters I trained a decision tree regressor on the first 70% of the data and tested it on the remaining 30%. This gave a training error (mean absolute error) of 16.4 and a validation error of 20.0.

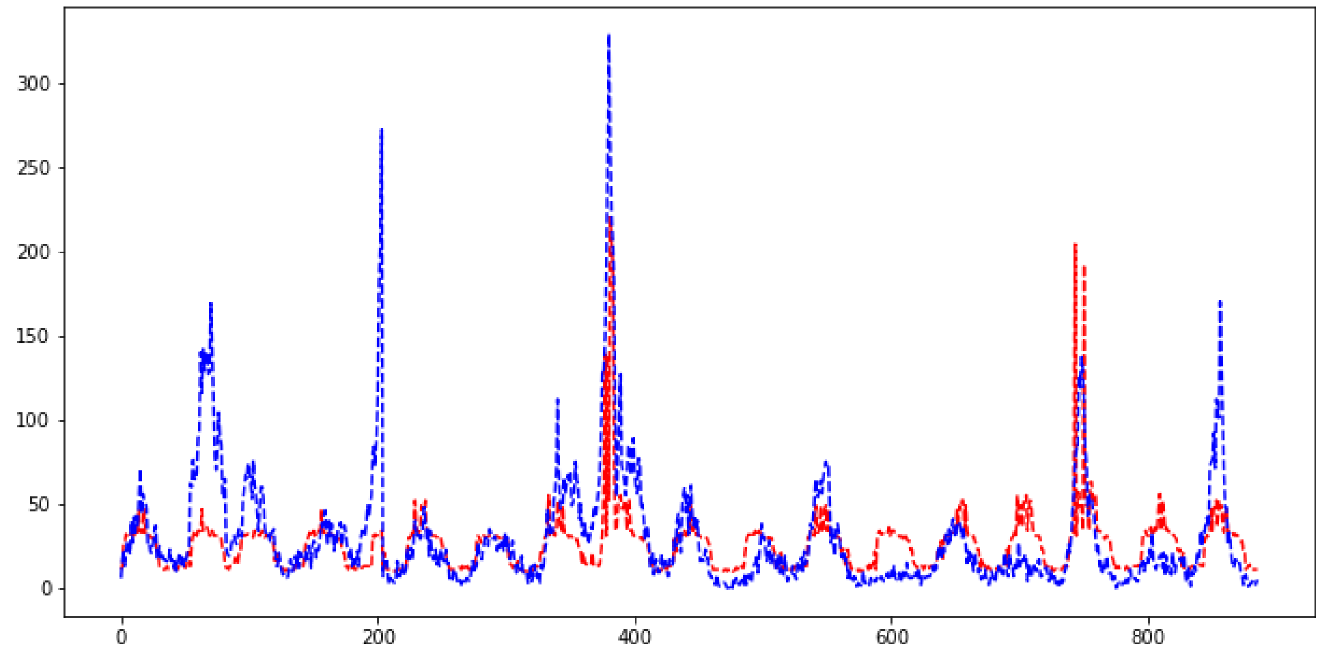
A plot of this regressor's predictions versus the label data shows that it has a good handle on the seasonal variation and sometimes predicts one of the major outbreaks. The predictions are quite noisy, even more than the actual data.

Decision Tree Regressor Predictions. First 620 points are training samples. The rest are validation



Using these parameters for the decision tree regressor, I used this as the weak learner in an Adaboost ensemble. I did a grid search again to see what sort of parameters would work best, trying n_estimators values of 3,10,20,40 and learning_rate of 0.02, 0.1, 0.3, 1.0. This suggested the best parameters were 20 learners and a  learning rate of 0.02. Using these parameters an Adaboost regressor was trained, again on the first 70% of the data and then tested on the final 30%. This gave a training error of 16.5 and a validation error of 15.6. This is only marginally better that the original Decision tree regressor. A plot of the predictions show that they are less noisy than for the single decision tree, but are otherwise broadly similar. Both the simple Decision Tree and the Adaboost ensemble performed better on the validation set than the training set. I think that this is just down to the fact that validation set had smaller outbreaks, which the models do not predict well. Overall however the estimator seems to perform better than the benchmark which has a validation error of 22.

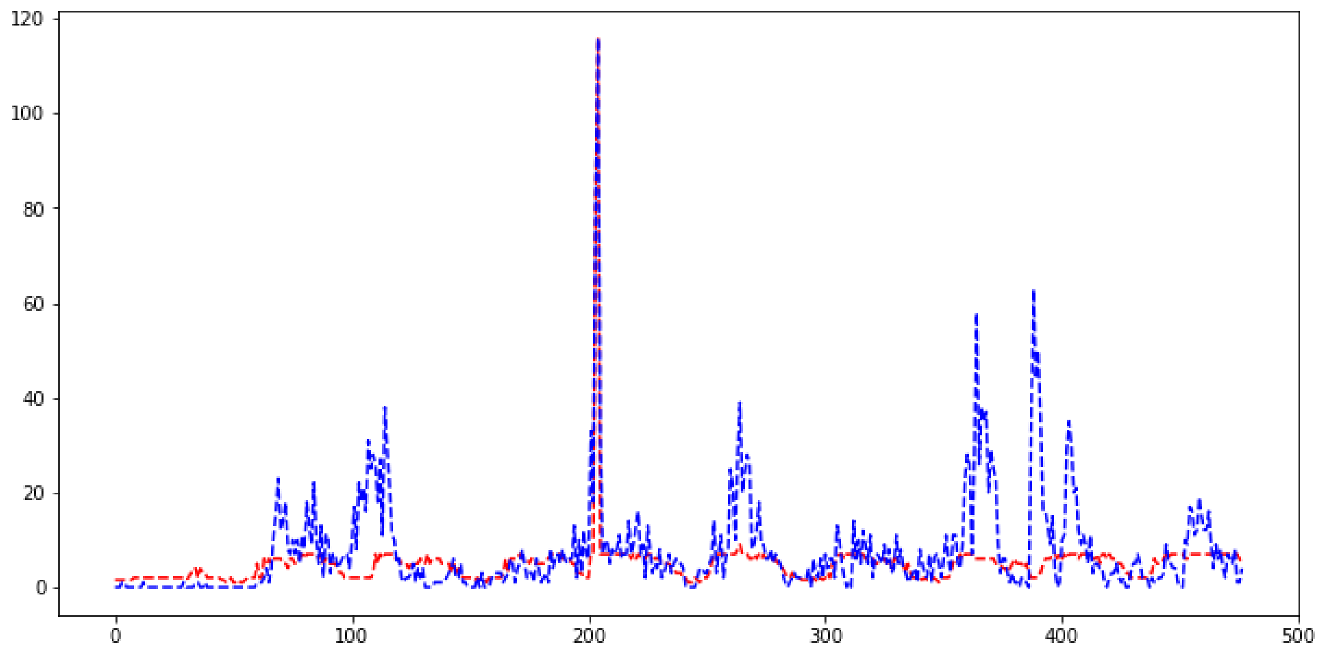
Adaboost Predictions for San Juan. First 620 points are training samples. The rest are validation



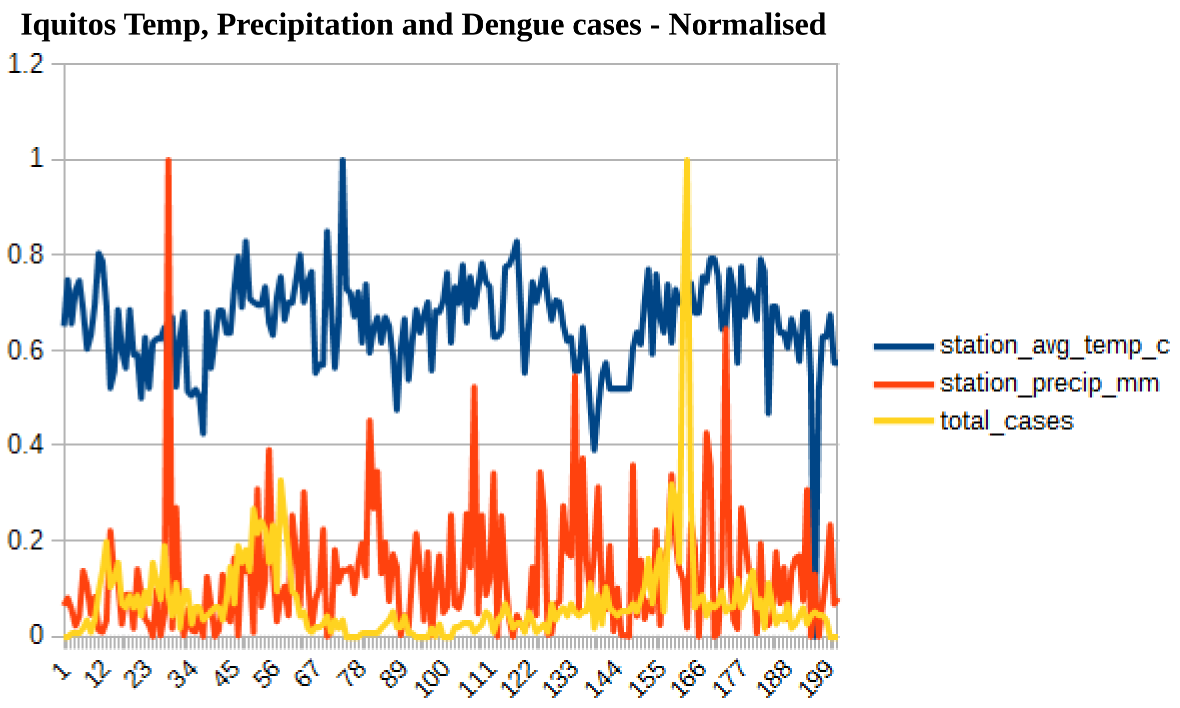
Moving on to the Iquitos data, I followed mostly the same steps. I did not remove the ndvi_ne feature as it did not have the large number of missing values as the San Juan data. I considered removing the first 70 data rows because the number of reported dengue fever cases during this period looked anomalously low, with no more than one case reported in any week during that period. However there are other periods in the data with very low reported cases, this is just are rather longer period. On the whole there didn't seem to be any clear evidence that these data points were erroneous so I left them in. I created the time shifted data and again removed incomplete data rows caused by this process. I used the same grid search to find the best parameters for a Decision Tree Regressor and used this as the week learner for the ensemble stage. I ran the grid search over the number of estimators and learning rate for the Adaboost method. This gave different parameters than for the San Juan data, with 20 estimators (as before) but a much higher training rate of 0.3. Using the resulting best estimator, trained on the first 70% of the data and then validated against the remaining 30% gave a training error of 3.9 and a validation error of 7.3.  Below is a plot of the Iquitos predictions (red) and actual cases (blue). As before the first 70% or 364 weeks are the training set and the rest are the validation set. The estimator has had some success predicting one of the outbreaks, but overall is no better than the benchmark model. In fact, given that the mean absolute deviation of the data itself is 6.68, the model is actually worse than just predicting the average!

The Iquitos data seems to be much more difficult to model. The lower number of cases means that the data is somewhat more noisy which may be part of the reason. There also does not seem to be much, if any, seasonal trend.

**Adaboost Predictions for Iquitos. First 364 points are training samples. The rest are validation**



Below is a plot of average temperature, station precipitation and fever cases for Iquitos, similar to the plot for San Juan earlier in this report. The seasonal temperature cycle is visible but much less clear than for San Juan. The precipitation is similarly noisy.



## Assessment

To be honest the results are a little disappointing. Although the San Juan validation score was a considerable improvement over the benchmark I think there was some luck in this. During the testing of the model I also looked at some cross-validation scores which were as follows,

```
cv scores:[-30.48, -17.38, -8.78, -42.10, -12.51,-11.45, -14.65, -16.07]
```

So the model is quite sensitive to which data is used for training and validation. The model has managed to predict the seasonal trend quite well but is not very good at predicting the outbreaks. This is also reflected in its R2 score of 0.22 for the validation set and 0.35 for the training set.

As for the Iquitos model, on the mean absolute error metric this was worse than predicting the mean value for all data points. The R2 score of -0.15 for the validation is equally poor.

Anecdotally I tried some manual adjustments of some model parameters. Increasing the maximum depth of the regression trees clearly enabled the model to train better, but this just led to worse performance on the validation set – i.e. over-fitting. For example increasing the maximum depth to 6 reduced the training error to 9.9, but the validation error increased to 18. Also increasing the number of estimators made matters worse.

## Refinement

The model developed so far seems to be somewhat limited. It can predict the seasonal trend (at least for San Juan) but attempts to improve on this by increasing the model complexity lead to over-fitting and a worse score on the validation set. I think the problems are three fold:

1) The data is very noisy. Attempts to improve the predictive power of the model lead it to train on the noise, reducing its ability to generalise

2) I have a lot of features and many are very poorly correlated with the number of dengue cases. Having said that, if I had features that were strongly correlated to dengue cases then I would not need a machine learning algorithm. I think the problem, again, comes back to the noise in the data.

3) The problem is inherently one of population dynamics, which are well known to be chaotic and highly non-linear; and therefore very difficult to predict.

I think my most likely route to a better prediction lies with better pre-processing. As the model has a strong tendency to over-fit, I will start by reducing the dimensionality using principal component analysis.

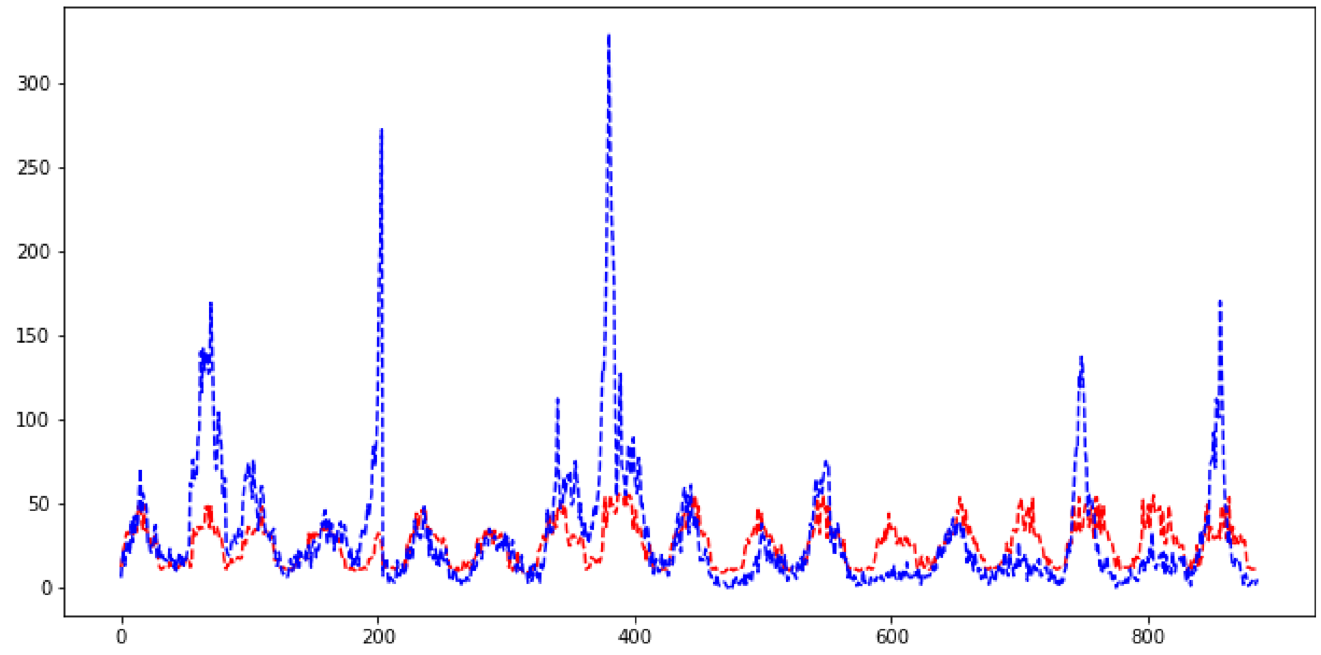
Using principal component analysis I reduced the number of features to 6 and re-ran the model optimisation. This time the optimum tree depth was 4 with a minimum split on 2 data points. The results of this model were however, worse than for the previous model. The training and validation error were both 18.5.

I tried increasing the number of features to 10 and 20, but without significant improvement. At 20 features the training error was 17.8 and the validation error 19.3. In general none of these models is doing much more than predicting the seasonal variation. With 20 features if I force the max depth of the decision trees up to 10 I get the training error down to 2.8, but the validation error goes up to 20.6.

Some articles I have read suggest that the Adaboost ensemble method is more sensitive to noise than some other methods. So I tried a random forest. I used a cross validated grid search with a random forest regressor to explore a range of maximum tree depths, namely [2,3,4,6,10] and a range of n_estimators of [10,20,40]. The best results were found with a maximum depth of 2 and with 40 estimators. This gave a training error of 18.9 and a validation error of 19.2. Looking at a plot of the predictions, again the model is just predicting the seasonal variation.

At this point I am trying to decide whether it is best to go for a different type of estimator, or to try to create some new features that might help the existing model. Something I am conscious of is the fact that mosquito populations would be expected to grow exponentially when the conditions are right. Perhaps creating some new features that are exponentially related to the measured features would help the model. First I will reduce the number of features by removing those for which there is a "reanalysis" of the same underlying parameter, or where there is clearly another feature that carries the same information. Then I will add the time shifted features, as before, and finally add exponentiated versions of all these features. To avoid the exponentiation leading to overflows I will need to scale the data first.

Below is a plot of the resulting predictions for San Juan. The validation error for these predictions was 16.8, while the training error was 17.1



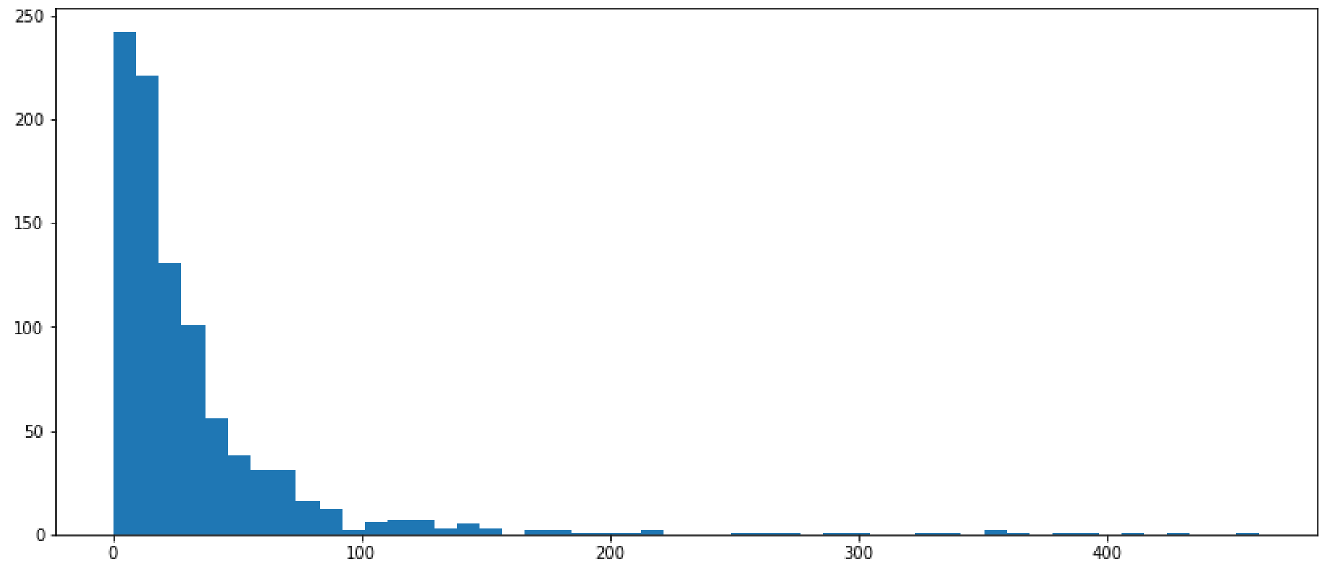
## Conclusion

I am somewhat disappointed in my results. For the San Juan case I was able to do somewhat better than the benchmark, but still only really predicting the basic seasonal trend. The model was not predicting outbreaks, which would seem to be the real goal. For Iquitos the situation is worse with the model being worse than simply predicting 7.56 cases every week.

Returning to the beginning of the analysis, I think the large kurtosis figure for the label data is significant. It shows that most of the variance in the number of dengue cases comes from a small number of extreme cases, which might be characterised as outliers. But it is exactly these outliers that we would like to predict. Below is a histogram plot of the San Juan reported cases. The very long tail shows the issue. The models are learning from the majority of the data in the 0 to 80 range, but the amount of data in the tail is not enough for the model to learn on.

The Iquitos data shows a similarly large kurtosis. For Iquitos the seasonal temperature cycle is much less pronounced and there is a similar lack of seasonality in the cases data, making the problem even more difficult..

**San Juan Reported Dengue Cases Histogram**



## Reflection

This project has been very instructive in how tricky machine learning is. Over-fitting has been a constant barrier, with all attempts to increase the power of the models leading to better training at the expense of poorer generalisation. I also learned an important lesson about cross validation when consecutive data points are highly correlated, as they usually are in time series problems. An early cross validation scheme I used had the training and validation data sets interleaved in a fine grained way which effectively leaked all the information from the validation set into the training set!
Another challenge I have found is avoiding the tendency to try and “just tweak it to see if that makes it better”. A lot of time can be spent doing this, usually to little effect.

## Improvement

To move forward with this project it feels like I need a fundamental shift in approach. I think I should spend more time working on the data going into the models. I believe there are some techniques for transforming a data set that has high Kurtosis into something closer to Gaussian. Perhaps this would improve the models ability to learn the outbreaks without over-training. I had initially intended to try a Support Vector machine in addition to a decision tree approach, so I would consider my next step to be to find some techniques for dealing with high Kurtosis and then apply these, together with normal feature scaling, and train a support vector regressor on the resulting data.