Forecasting disease from univariate time series using naive, ARMIA, exponential smoothing, additive regression, and LSTM models

A capstone for Springboard’s data science bootcamp

Mike Pierovich

June 18, 2020

1. Introduction

Dengue fever afflicts more than 400 million people each year. For some, dengue fever is relatively mild. Symptoms include fever, headache, vomiting, and rash. Recovery takes a week. For others, dengue fever means severe bleeding, dangerously low blood pressure, and even death.

Dengue fever is a tropical disease, found around the globe. Dengue fever is transmitted to humans by mosquitoes carrying the dengue virus.

In this capstone, my hypothetical client is a non-governmental organization (NGO) focused on health care in Latin America. My client wants a multi-year forecast of weekly cases of dengue fever. The NGO will use the forecasts to allocate budget and deploy health-care resources. The NGO is focused on two cities--San Juan, Puerto Rico and Iquitos, Peru.

2. Data acquisition

I use data from DrivenData's data-science competition, "DengAI: Predicting Disease Spread."

- The data includes a count of weekly cases of dengue fever in San Juan and Iquitos. The data covers around 1,300 observations from the years 1990 through 2008.
- The data includes about 20 weekly climate variables, such as temperature and rainfall measurements. I chose to ignore the variables as my learning goal is to focus on univariate, not multivariate, time series.

3. Data assessment

The raw data is very clean as it is a data-science competition. For example, there are no missing values in the forecast variable of weekly cases. To further clean the data, I do the following: create a common datetime index; add a month column; reorder date-related columns; and split out the two cities.

4. Exploratory analysis

To start, I explore the weekly cases of dengue fever. I analyze the two cities independently. This is because San Juan, Puerto Rico and Iquitos, Peru are 1,600 miles apart. They have different climates and mosquito populations. The cities likewise have different (although similarly-sized) human populations and separate medical systems. Throughout this report, I focus on the larger San Juan data and, when appropriate, I compare how the Iquitos data differs from that of San Juan.

San Juan, Puerto Rico

- **Sample size, central tendency, and variance:** The San Juan data includes 936 observations of weekly cases over 17 full years and 2 partial years. The mean is 34 cases of dengue fever a week. The median is 19 cases. The standard deviation is relatively large at about 51 cases.

- **Distribution**: Weekly cases of dengue fever look to be more exponentially distributed than normally distributed as shown in figure 1. There is a large, positive skew. Most weeks have few or no cases. A few weeks have more than 200 cases. Figure 2 shows that the distribution of the log(x+1) of weekly cases looks a bit more normal.

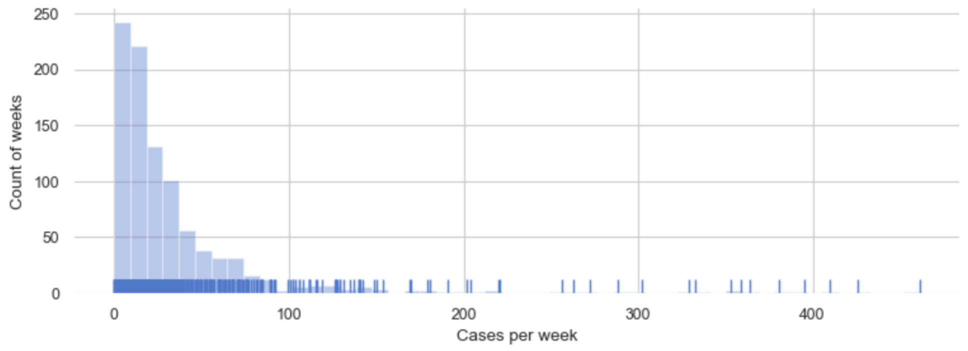


Figure 1: Histogram and rug plot of weekly cases in San Juan

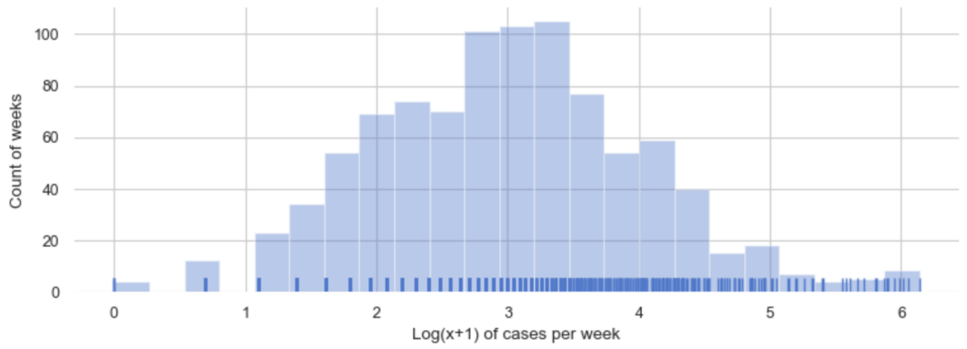


Figure 2: Histogram and rug plot of the log(x+1) of weekly cases in San Juan

- **Outliers:** The box plot of weekly cases in figure 3 highlights that most weeks have less than 50 cases. Yet many weeks have cases in excess of 1.5 times the interquartile range (IQR).

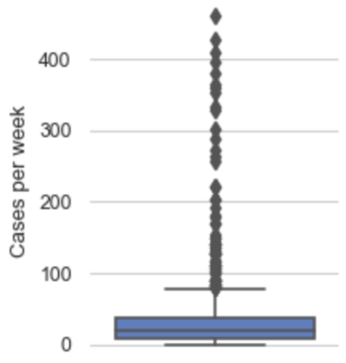


Figure 3: Box plot of weekly cases in San Juan

- **Weekly cases across all years:** These outliers become clear as "outbreaks" or "spikes" as shown in figure 4. The years 1991, 1994, 1998, 2005, and 2007 suffer from such spikes.

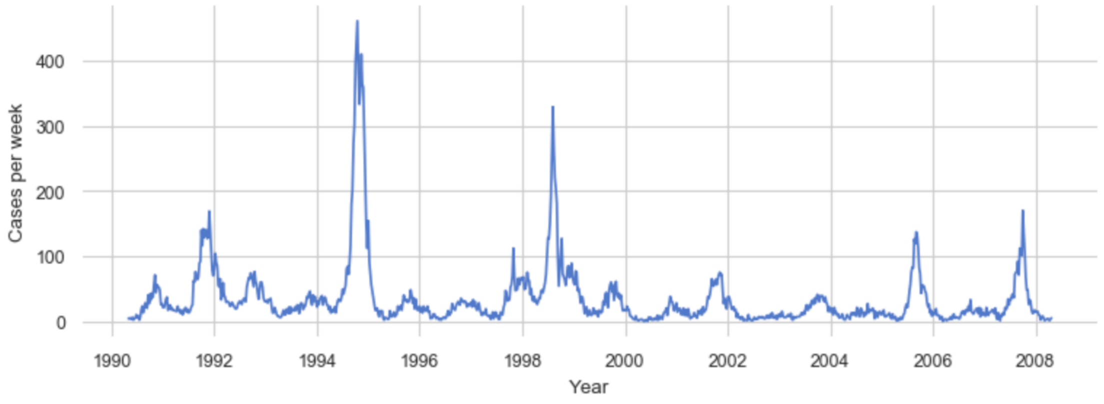


Figure 4: Time plot of weekly cases in San Juan

- **Weekly cases by year:** There's a slight trend of slowly declining weekly cases. This is hinted at in means cases by year as shown in figure 5. The box plot of weekly cases by year in figure 6 further highlights both the spikes and the slight year-over-year trend.

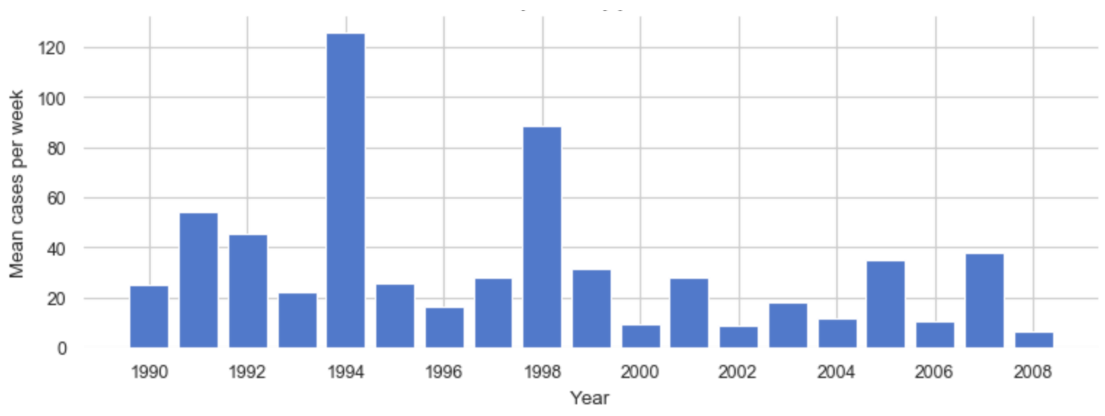


Figure 5: Bar plot of weekly cases by year in San Juan

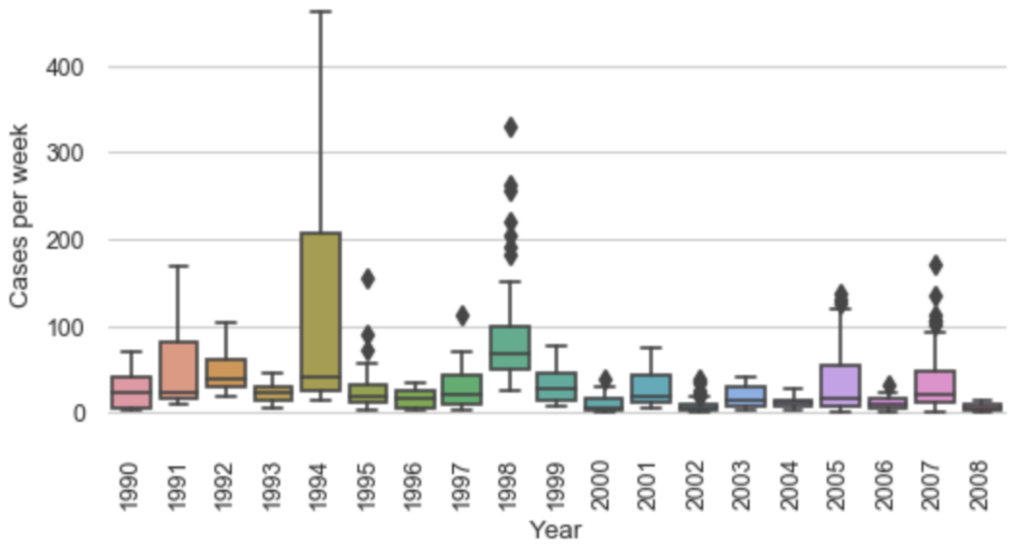


Figure 6: Box plot of weekly cases by year in San Juan

- **Weekly cases by month:** The bar plot of average cases by month in figure 7 shows the seasonal pattern of dengue fever. Cases peak in November at 71 per week and lessen in April to 10 per week. The box plot in figure 8 shows that San Juan's fever season, which is from August through December, has a higher mean, wider IQR, and numerous spikes.

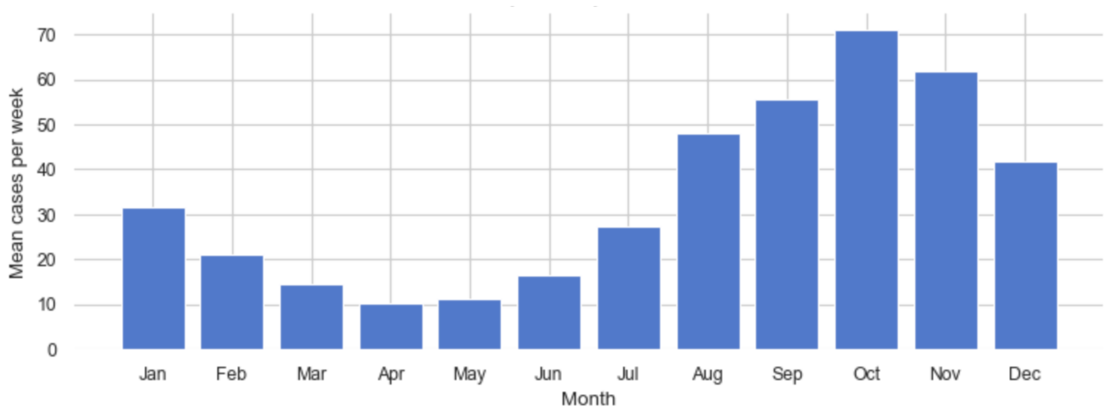


Figure 7: Bar plot of average weekly cases by month in San Juan

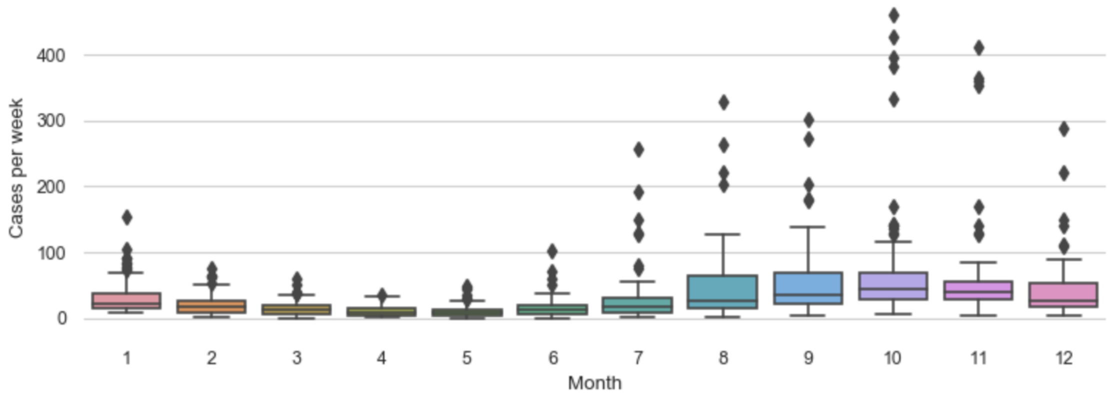


Figure 8: Box plot of weekly cases by month in San Juan

- **Trend, seasonality, and residuals:** I decompose the time series into trend, seasonality, and residuals using an additive model from the statsmodels package. See figure 9. The trend does not have a clear direction, maybe a slight downward movement. The predominant trend features are the larger "humps" (corresponding to the terrible years of 1994 and 1998) and smaller "humps" (corresponding to the bad years of 1991, 2004 and 2005). There is strong seasonality within the year. The residual indicates the naive additive model isn't capturing all information in the data. For example, the residual does not appear to be random, to have a mean of zero, or to have a constant variance.

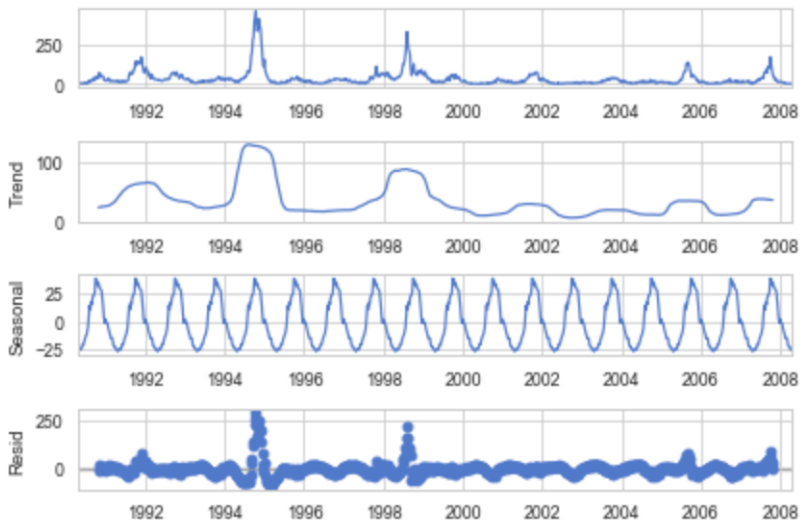


Figure 9: Decomposition of weekly cases
into trend, seasonality and residual
via a naive, additive model in San Juan

Iquitos, Peru

The pattern of dengue fever in Iquitos is generally similar to that of San Juan. However, there are a few notable differences.

- **Smaller sample**: The Iquitos data sample is smaller, covering one, not two, decades.
- **Fewer cases**: Iquitos has fewer cases--about ¼th of the weekly cases on average.
- **Smaller spikes**: Iquitos suffers from outbreaks or spikes in cases as shown in figure 10--although these spikes may be less severe.

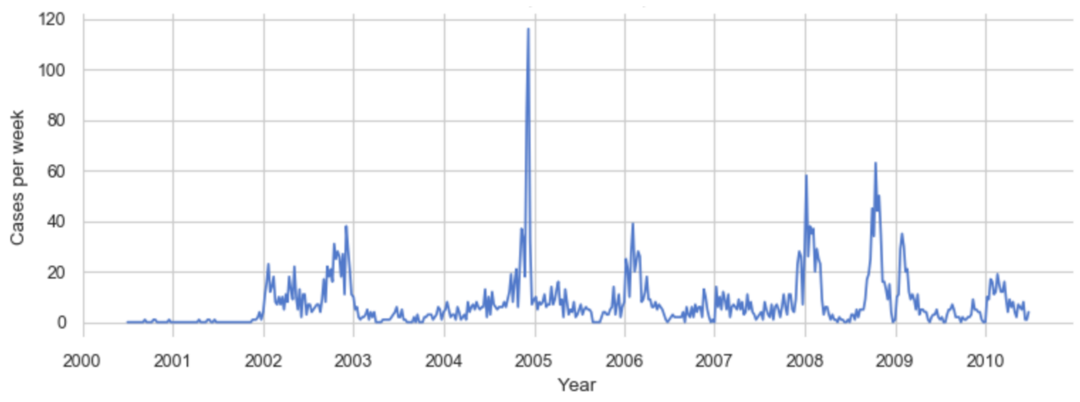


Figure 10: Time plot of weekly cases in San Juan

- **Shifted seasonality:** The seasonality of dengue fever in Iquitos is different from that of San Juan. Iquitos' season is shifted about two months later in the year. Iquitos' high-fever season starts in October, peaks from December to February, and declines in March as shown in figure 11.

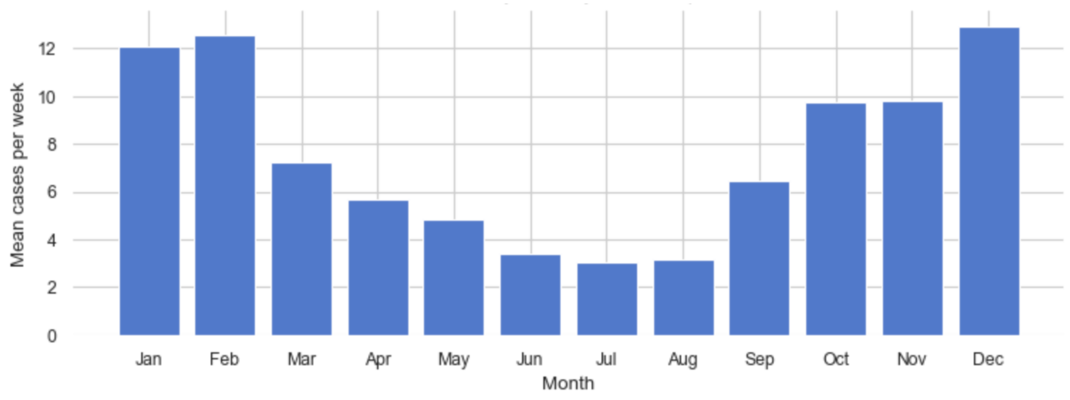


Figure 11: Bar plot of average weekly cases by month in San Juan

5. Statistical analysis

Next, I analyze the autocorrelation and stationarity of weekly cases of dengue fever. I find that the data is autocorrelated and is not stationary, in both San Juan and Iquitos.

San Juan, Puerto Rico

- **ACF**: A graph using statsmodels' autocorrelation function (ACF) shows a strong, gradually-declining correlation as time lags increase. See figure 12. This points to a slowly declining trend. Plus, the change from a positive to a negative correlation at about week 20 is consistent with seasonality.

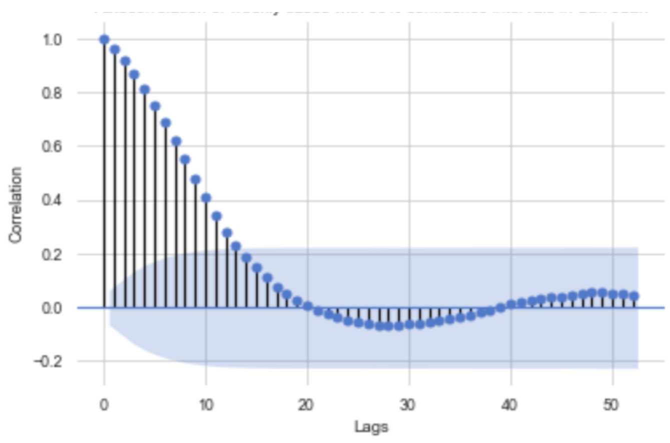


Figure 12: Autocorrelation of weekly cases with 95% confidence interval in San Juan

- **PACF**: A graph using the partial autocorrelation function (PACF) highlights the strongest autocorrelation is with week 1. See figure 13. This also suggests a weak negative correlation with weeks 2, 3, 5, 6, and 9.

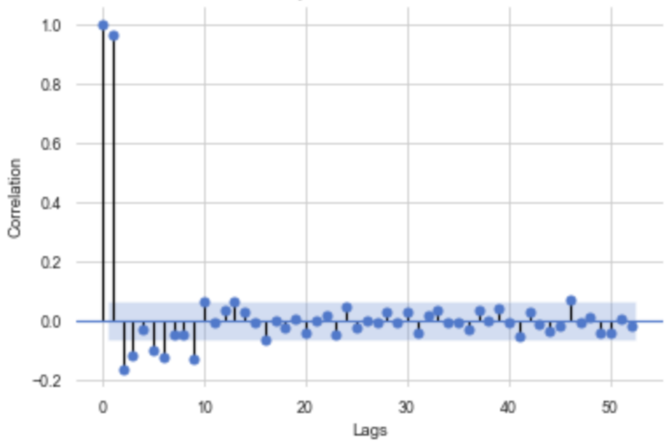
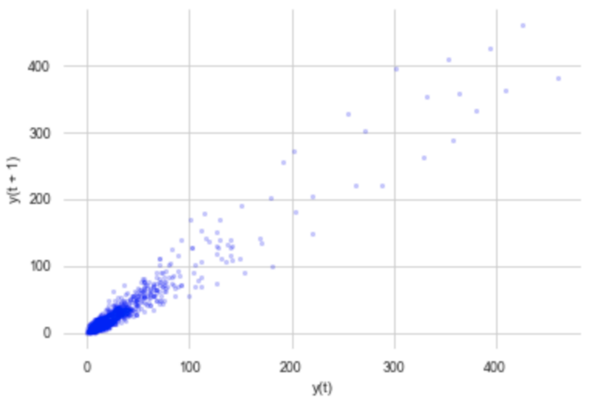
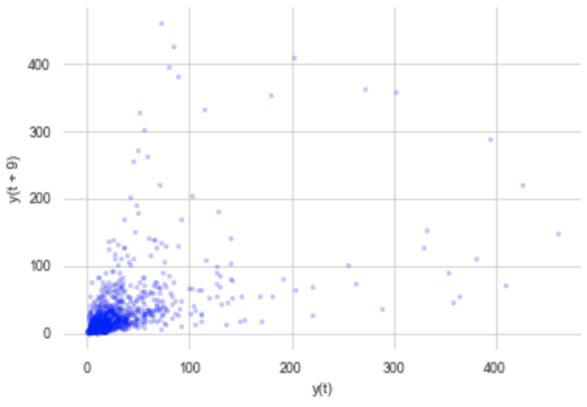


Figure 13: Partial autocorrelation of weekly cases with 95% confidence interval in San Juan

- **Lag plots**: Lag plots validate some autocorrelation at week 1 and 2, but hint at less in later weeks. For example, figure 14 shows the stronger 1-week lag and figure 15 shows the weaker 9-week lag.





Figures 14 and 15: Lag plots against week 1 and weeks 9, respectively, in San Juan.

- **Not stationary:** Weekly cases of dengue fever in San Juan are not stationary. To reach this conclusion, I do the following:
  - Apply some "down-n-dirty" tests of examining the mean and variance of the first and second halves of the data. The differences in these metrics support a conclusion of "not stationary".
  - Inspect the decomposition of the data into trend, seasonality and residuals. The clear seasonality supports "not stationary."
  - Apply two hypothesis tests for unit roots--the Augmented Dickey-Fuller (ADF) test and the Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test. Both of these tests point towards statistically-significant "stationary." For me, the graphs and inspection outweigh the unit root tests.

Iquitos, Peru

Weekly cases in Iquitos are autocorrelated and not stationary as well.

- Iquitos' ACF is roughly similar to San Juan's, although Iquitos has two patches of negative correlation.
- Iquitos' PACF is likewise roughly similar to that of San Juan, although Iquitos shows positive correlation at weeks 1 and 2 and negative correlation at week 5.
- Iquitos' lag plots at weeks 1, 3, and 5 indicate that the correlation isn't too strong.
- Comparing summary statistics and visual inspection point to not-stationary data despite the fact that ADF and KPSS tests point towards statistically-significant stationary data.

6. Data preparation

To prepare for forecasting, I clean, transform, and split the data.

- **Clean:** With my focus on univariate forecasting, I drop all columns other than the date (i.e., my index) and the forecast variable (i.e., weekly cases). Also, I drop some “suspect” values in the Iquitos data. These values are Iquitos’ first two years of observations. These observations are mostly zeros, show little variance, have no seasonality, and, right at the start of the 2002 calendar year, move to a typical pattern. Something seems off.

- **Transform:** Often, time-series models perform better with stationary data. I transform my non-stationary data with differencing, logs, and Box-Cox. I evaluate whether the transformed data is stationary or not. To do so, I use summary statistics, seasonal decomposition, the ADF test, and the KPSS test. My transformed data appears to be stationary. The optimal lambda from the Box-Cox transformation is -0.035, which is very close to zero. This means Box-Cox is applying something close to a simple log transformation. Bottom line, I decide to use log(x+1) when I want to model with transformed data.

- **Split:** I split the time series into training and test sets with a 75%/25% allocation. I use the "earlier" part of the series for training and the "later" part for testing as shown in figure 16.

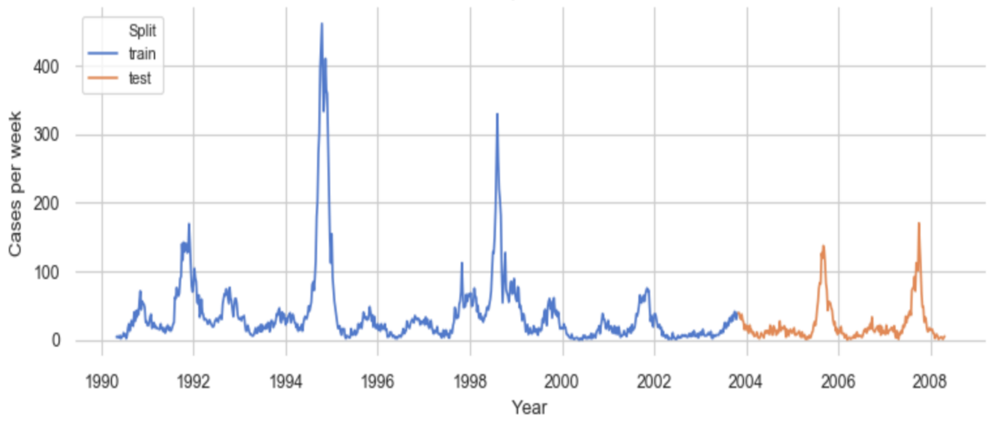


Figure 16: Train/test split for San Juan

7. Forecasting methods

I use five approaches to forecast weekly cases of dengue fever. In this section, I summarize these approaches, share performance metrics, assess the best performers, and share some observations.

Approaches

I tackle this work as a time-series problem. This is because of the data, the domain, and my learning goals. The data is a temporal series (i.e., decades-long sets of weekly observations) with strong autocorrelation. Autocorrelation is also a feature of the domain (e.g., infections from last week drive infections this week). And my learning goal is to better understand time-series forecasting.

My approaches for time-series forecasting are: naive, ARMIA, exponential smoothing, additive regression, and LSTM models. I score each approach with mean absolute error (MAE) and root mean square error (RMSE). I use MAE because it is easily interpretable and is used in this data-science competition. I use RMSE because of its sensitivity to larger errors. Generally, I select hyperparameters using the San Juan data and apply San Juan's best performing hyperparameters to Iquitos. Below are summaries of each approach.

- **Naive:** These are the simplest approaches to forecasting that I use as baselines. With the **naive method**, future values are forecast as the last observed value (e.g., next week's forecast value is this week's observed value). With the **seasonal naive method**, future values are forecast as last season's observed value (e.g., next week's value is the observed value from 52 weeks before, assuming annual seasonality). With the **average method**, future values are forecast as the mean of historical values (e.g., next week's value is the mean of prior observed values). The window of historical values used to calculate the mean can vary from just a few recent values to all values.

- **ARIMA**: This classic time-series method focuses on the autocorrelations in the data. To implement this approach, I use the Pyramid auto_arima package, which is based on a prior R function. Auto_arima fits both ARMIA-type and seasonal or SARIMA-type models. Auto_arima grid-searches across key parameters (e.g., ARIMA's p, d, q and SARMIA's P, D, Q and m), while minimizing the Akaike information criterion (AIC). I use mostly default settings, other than providing the yearly seasonality (e.g., m=52).

- **Exponential smoothing:** With this classic time-series method, forecasts are weighted averages of past observations with the weights decaying exponentially as the observations get older. Of the many exponential smoothing (ES) models, I use three. **Simple ES**, which is also called single ES, focuses on the series' level. **Trend ES**, which is also called double ES and the Holt method, extends ES to account for a trend. **Seasonal ES**, which is called triple ES or the Holt-Winters' method, further extends ES to account for seasonality. These methods have one, two or three key parameters, respectively, which adjust the weight between current and past observations. Trend ES and Seasonal ES can be additive or multiplicative, which describes how the variance of the trend or season changes over time.

- **Additive regression:** Additive models are a type of nonparametric regression that use one-dimensional smoothers to restrict generalized regression models. These models are thought to achieve a balance in flexibility, overfitting, data requirements, and interpretability somewhere between linear regression and generalized additive models. This approach can be thought of more as a curve-fitting problem than an examination of time-based dependencies. To implement this approach, I use Prophet (FB Prophet), an open-source, business-focused forecasting tool created by Facebook. This tool uses additive regression to separately model level, trend, seasonality, and holidays.

- **LSTM:** Long short-term memory models (LSTM) are deep-learning models based on recurrent neural network (RNN) architectures. LSTM networks learn temporal dependencies and forecast time series. LSTM's architecture of cells, input gates, output gates and forget gates allow LSTM to read each observation and build up representations used in prediction.

  I work through five types of LSTM. The **simple LSTM** has one hidden layer of LSTM units and one output layer. The **stacked LSTM** has multiple layers of hidden LSTM. The **bidirectional LSTM** learns the sequence both backward and forward. The **CNN-LSTM** is a hybrid that first processes data through a convolutional neural network (CNN) and then through a simple LSTM. The **ConvLSTM** is another hybrid model that builds the CNN-related layer into the LSTM layer.

Performance

The performance of these approaches and their variations is shown for San Juan in table 17.

Approach	Variation	Transform	RMSE	MAE	Rank
Additive Regression	Stabilized Trend	Log (x+1)	28.70	14.59	1
Additive Regression	Cap and Floor	Log (x+1)	29.94	14.64	2
Exponential Smoothing	Seasonal ES, a=.5, b=.1, g=0, optimized	Log (x+1)	26.54	15.54	3
ARIMA	SARIMAX (2, 1, 2) x (2, 0, 1, 52)	Log (x+1)	26.61	15.85	4
Exponential Smoothing	Seasonal ES, a=.9, b=.8, g=.1, optimized	None	28.60	17.14	5
ARIMA	SARIMAX (3, 1, 2) x (0, 0, 0, 52)	None	29.48	18.71	6
Exponential Smoothing	Simple ES, a=0	None	34.12	19.09	7
Additive Regression	Stabilized Trend	None	29.20	20.05	8
Naive	Seasonal Naive Method	None	35.95	21.34	9
LSTM	ConvLSTM	Log (x+1)	34.29	21.36	10

LSTM	ConvLSTM	None	35.03	22.49	11
LSTM	CNN-LSTM	None	35.05	22.75	12
Additive Regression	Cap and Floor	None	30.87	23.43	13
LSTM	Stacked LSTM	None	36.53	24.49	14
Exponential Smoothing	Simple ES, a=.2	None	30.28	24.52	15
Exponential Smoothing	Trend ES, a=.2, b=.8, dampen	None	30.28	24.52	16
LSTM	Stacked LSTM, 2.5 times nodes	None	36.89	25.02	17
LSTM	Simple LSTM	None	37.16	25.13	18
Exponential Smoothing	Simple ES, a=.8	None	32.11	27.61	19
Naive	Average Method	None	32.39	28.03	20
Naive	Naive Method	None	32.87	28.71	21
Exponential Smoothing	Simple ES, a=1.0, optimized	None	32.87	28.71	21
Exponential Smoothing	Trend ES, a=.8, b=.2, dampen	None	33.28	29.27	23
LSTM	Bidirectional LSTM	None	43.22	30.30	24
Additive Regression	Default	None	40.39	30.96	25
Exponential Smoothing	Trend ES, a=1, b=0, optimized	None	36.07	32.81	26
Exponential Smoothing	Trend ES, a=.8, b=.2	None	170.64	151.08	27

Table 17: Performance of models and variations, ranked by MAE, for San Juan

The performance of the Iquitos forecasts is shown in table 18.

Approach	Variations	Transform	RMSE	MAE	Rank
Additive Regression	Stabilized Trend	Log (x+1)	11.72	6.52	1
ARIMA	SARIMAX (1, 0, 1) x (0, 0, 0, 52)	Log (x+1)	11.95	6.76	2
ARIMA	SARIMAX (1, 0, 4) x (2, 0, 0, 52)	None	11.02	7.02	3
LSTM	ConvLSTM	Log (x+1)	12.51	7.44	4
Naive	Average Method	None	11.26	7.65	5
Exponential Smoothing	Seasonal ES, a=.4, b=.4, g=.2, optimized	Log (x+1)	12.99	7.85	6
Exponential Smoothing	Seasonal ES, a=.6, b=0, g=0, optimized	None	12.83	8.15	7
Naive	Naive Method	None	13.79	8.16	8
Naive	Seasonal Naive Method	None	16.48	9.92	9

Table 18: Performance of models and variations, ranked by MAE, for Iquitos

Assessment

These models vary in their performance. In this section, I examine how these approaches performed on the San Juan data.

- **Best of Naive**: Of the naive approaches, the seasonal naive method performs the best on the San Juan data. This makes sense as this model is picking up on the seasonality in the data while the other methods produce “flat” forecasts. A look at the residual of this model--including the mean, distribution, Q-Q plot and ACF plot--suggests there is bias and the forecast can be improved.

- **Best of ARIMA:** The best of ARMIA is a seasonal model (i.e., SARIMA) on log-transformed data. The key parameters for San Juan are: p of 2, which represents 2 autoregressive terms; P of 2, which represents 2 seasonal autoregressive terms; d of 1, which represents 1-lag ARIMA differencing; D of 0, which means no seasonal differencing; q of 2, which represents 2 ARMIA moving-average terms; and Q of 1, which represents 1 seasonal moving-average term. This model scores better than all naive models and has an MAE of about 16. The model also succeeds in capturing the seasonality as shown in figure 19.

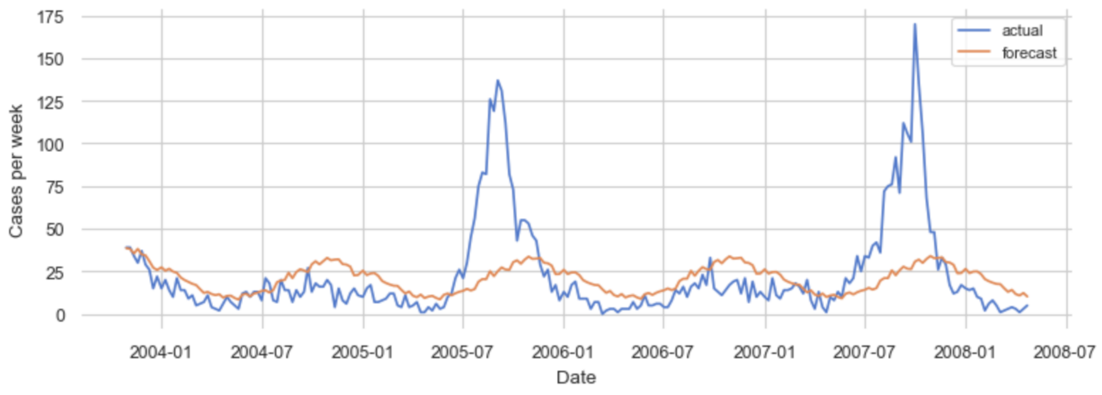


Figure 19: Actual and forecast of SARMIA model on log(x+1) data for San Juan

- **Best of exponential smoothing (ES):** The best of the ES is seasonal ES on log-transformed data. This model's key parameters are: an alpha or level smoothing of about .47, which means the model is using some, but not all, historical data for the level; a beta or trend smoothing of about .05, which means the model is using almost all historical data for the trend; and a gamma or seasonal smoothing of 0, which means the model is using all historical data for the seasonality. This model outperforms the naive and ARMIA models and is shown in figure 20.

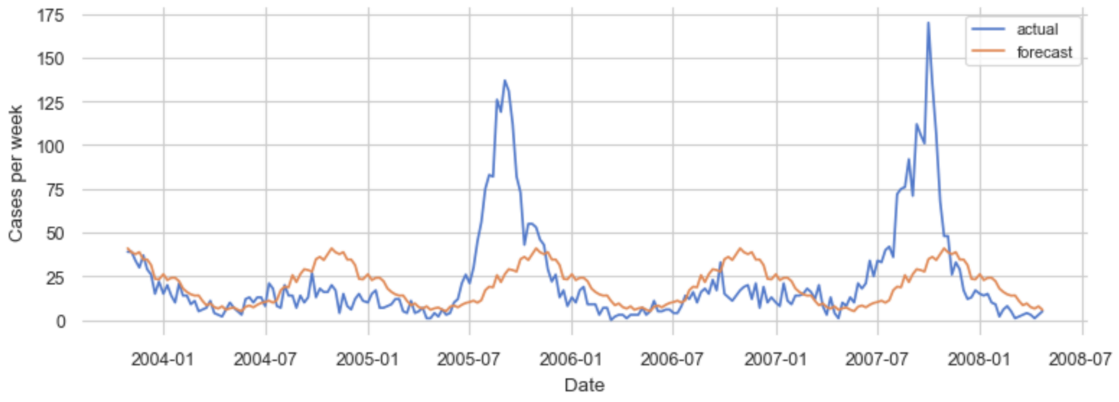


Figure 20: Actual and forecast of Seasonal ES model on log(x+1) data for San Juan

- **Best of additive regression:** I work through three variations of this approach with FB Prophet. These include a "default" variation, a "stabilized trend" variation (i.e., a less-responsive trendline with fewer shifts in direction) and a "cap and floor" variation (i.e., parameters to prevent non-negative forecasts). The stabilized-trend variation on log-transformed data performs best. This forecast has a slowly declining trend and a "smooth" seasonality as shown in figure 21.

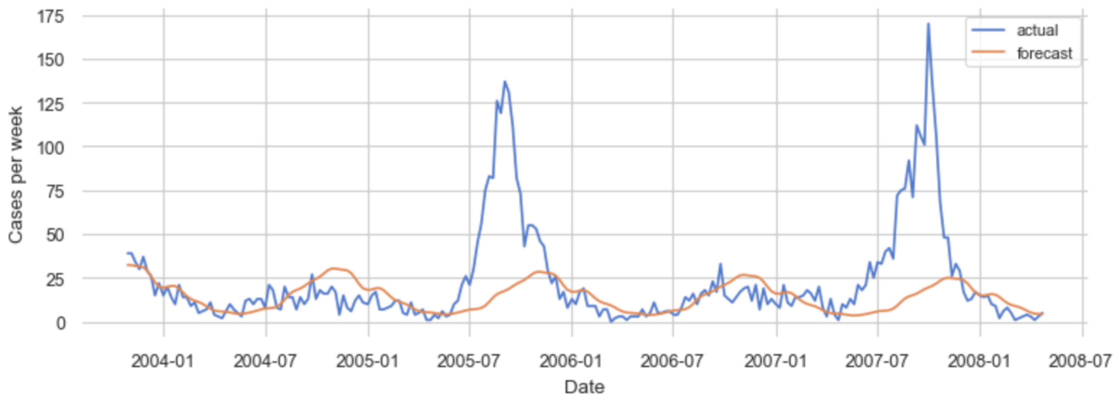


Figure 21: Actual and forecast of an additive model using FB Prophet with "stabilized trend" arguments on log-transformed data for San Juan

- **Best of LSTM:** Of the LSTM variations, ConvLSTM on log-transformed data performs best. The second-best performing is CNN-LSTM. The worst performing model is bidirectional LSTM. Interestingly, these LSTM models don't produce the regular "seasonal" curves of the classic time-series methods, as shown in figure 22.

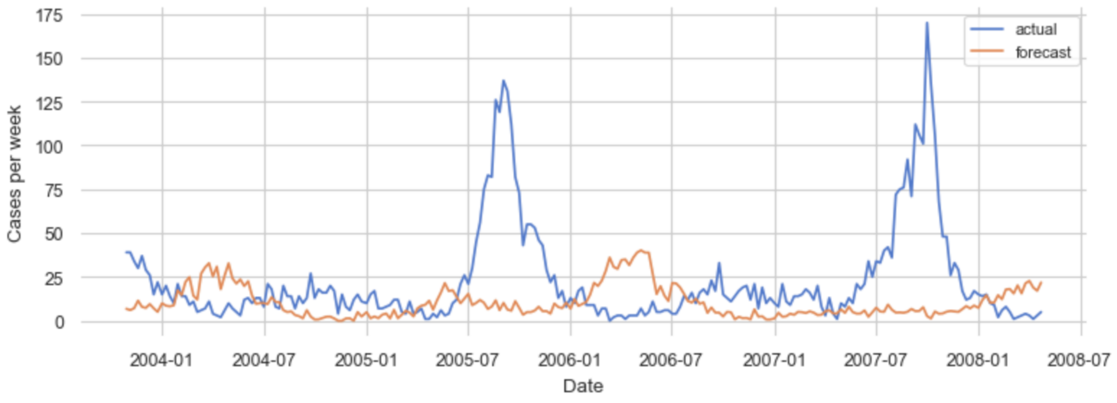


Figure 22: Actual and forecast of ConvLSTM on log-transformed data for San Juan

- **Best of all:** The "stabilized trend" additive model performs best of all. In general terms, the approaches rank: additive regression, exponential smoothing, ARMIA, naive, and LSTM.

Observations

Here is what I learn while modeling:

- **Importance of well-transformed data:** The models using log-transformed data outperformed their counterparts using non-transformed data. Similarly, some of these models produced terrible results until I pre-processed the data through scalers. This makes sense as transforming and scaling can simplify the data, normalize it, and reduce variance, improving model performance. This highlights the importance of fully exploring transformations and understanding the properties of the transformed data. For example, I elect to use log-transformed data, rather than Box-Cox transformed, mainly for my convenience, which might have reduced performance.

- **Connection between data and model:** The basic character of my data is a slight trend, strong annual seasonality, and very strong periodic "spikes." The additive models used by FB Prophet are explicitly built upon fitting time series to trend and seasonal components. This connection between the data and the model helps me understand why the additive models performs best.

- **Limits of univariate.** The classic time-series approaches--ARIMA, exponential smoothing, and additive regression--on univariate data captured the seasonal nature of my data. They failed, however, to recognize the spikes in cases (and this isn't surprising as such spikes aren't the focus of such models). Nevertheless, predicting these spikes is a key part of this forecasting problem. I suspect the best clues to these spikes are not to be found in the univariate data, rather such indicators are to be found "external" variables, such as climate and temperature variables (i.e., things that have a big influence on the mosquito populations).

- **LSTM's lack of success:** I am surprised that LSTM is my worst-performing approach and was beat by even a naive forecast. I attribute this poor performance to a lack of data, which can cripple many deep-learning models. For me, the lack of data becomes all the more clear when I realize that my simple LSTM, for example, is being trained on a sequence of 701 numbers (i.e., an X of 9 years of weekly cases plus a y of another 4.5 years of weekly cases). My data, which is a sequence of 934 numbers, has only 234 such sequences for the LSTM to learn from. This isn't much data at all!

8. Final forecast

In this section, I discuss my final forecast.

- **Request for the final forecast:** I use the data-science competition's submission requirements as a proxy for my client's requests for a final forecast. Accordingly, the client asks for a 260-week (i.e., a 5 year) forecast of weekly cases of dengue fever in San Juan and for a 156-week (i.e.,a 3-year) forecast for Iquitos. This is a request for quite a long-term forecast.

- **Final model:** To create these forecasts, I use the best model (i.e., the additive "stabilized trend" model off log-transformed data using FB Prophet), retrain it on all sample data (e.g., pre-split data) for each city, and predict out the required periods. Then, I invert the scaling and transformations to get a final forecast.

- **Visualization:** Figure 23 is FB Prophet's visualization of the final forecast for San Juan. This figure includes the sample values (i.e., the back dots), the model (i.e., the darker blue lines) and the 80% confidence interval (i.e., the light blue area). In this figure, "y' is the log-transformed value of weekly cases.

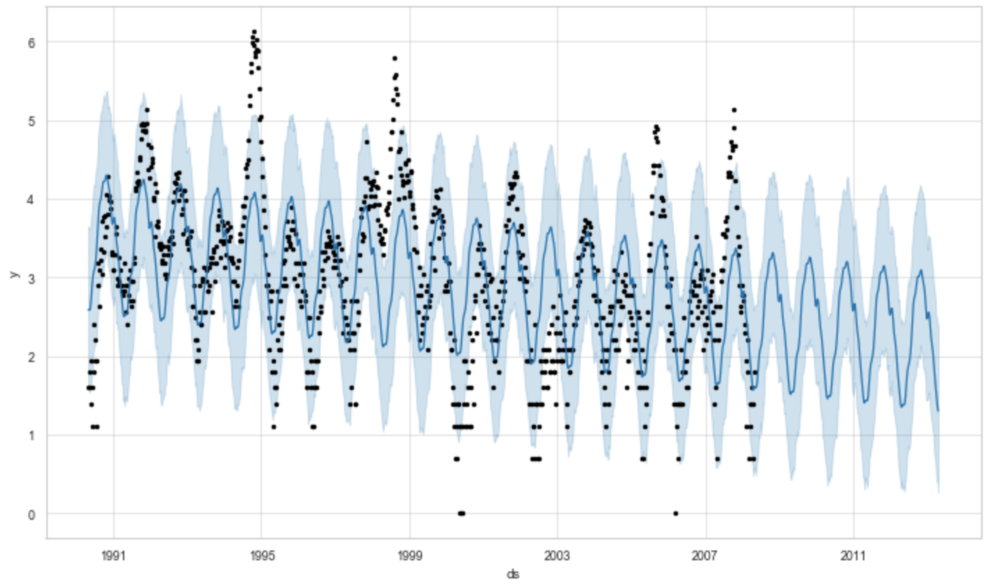


Figure 23: FB Prophet's visualization of the final forecast on a log(x+1) scale for San Juan

- **Confidence interval**: FB Prophet calculates an 80% confidence interval, including a lower bound and an upper bound for each forecasted value. The tool calculates this interval from the uncertainty in the additive model's trend component and from the noise in the observed values. The tool excludes uncertainty from the model's seasonal component, which, in reality, is likely a very big contributor. Plus, the model assumes that uncertainty is constant over time, not growing as forecasts extend into the future.

- **Example week with upper and lower bounds:** I pick a week that is 6-months into the 5-year forecast period for San Juan. For this week of August 19, 2008, the forecast value is 26 cases of dengue fever. The lower bound is 9 cases. The upper bound is 72 cases.

- **Score of final forecast:** As this is a data-science competition, I am able to score on my final forecast. The final forecast has an MAE of about 31. At the time of my submission, this forecast ranks at about 2,900 of 9,100 forecasts, and the top-10 leaders have MAE between 10 to 12.

- **Much bigger error than expected:** This result is much worse than I expect. My model likely has an MAE of around 12 when trained on my training set and scored on my test set. Yet the final forecast has more than two times my test MAE. Clearly, overfitting has crept into my modeling.

- **Wary of throwing out data:** The relative performance of my final forecast is worse than I expect. It ranks in only the top third of all entries. I suspect this stems from my “univariate” approach. From the outset, I chose to throw away the weekly observations of 20+ climate variables, which, most likely, are the key to predicting the spikes in dengue fever. If I were looking for a decent score, instead of learning about univariate time series, this is a terrible choice!

9. Next steps

I recommend the following next steps if permitted by time and circumstance:

- **Better understand the business context:** I would work with the client to better understand who would use the forecast, how they would use it, what actions they would take with the forecast and what are their organizational goals. For example: Are human resources types using it to hire? Are supply chain types using it to purchase?

- **Refine the forecast horizon:** I would work to understand how the forecast horizon helps solve the business problem, all in hopes of reducing the length of the forecast needed.

- **Better understand the domain:** I would better research how this disease is transmitted and how outbreaks occur (e.g., the mosquito lifecycle and mosquito-to-human transmission). This knowledge would likely influence model selection. In addition, I would speak with experts who are able to make “judgemental forecasts’’ and would try to understand the variables the expert considers important.

- **Expand univariate data:** I would look to expand my univariate data, looking for more data (i.e., more historical data), more fine-grained data (i.e., maybe daily cases), and broader data (i.e., more cities).

- **Leverage multivariate data:** I certainly wouldn’t limit myself to univariate data. Rather, I would explore multivariate data. And I would use those sources to better understand spikes.

- **Expand transformations:** I would explore additional transformations and adjustments, including additional mathematical transformations (e.g., a deeper exploration of power transformations) as well as additional calendar-, population-, and bias-focused adjustments.

- **Improve “programming” aspects:** I would continue to improve the “programming” aspects of my modeling. In my later notebooks, I better leverage python functions, write assert statements, write .py files for cross-notebook functionality, and leverage unit tests. I would continue to improve these techniques.

- **Explicitly predict spikes.** I would search for models--or even ensemble approaches--that enable the forecast of outbreaks in dengue fever.

- **Expand time-series models:** I might expand my time-series models to include dynamic regression models, hierarchical models, and complex seasonal models. In addition, I might further explore the machine-learning and deep-learning aspected of this problem, especially if I am able to obtain larger data sets.