CHAPTER 1
INTRODUCTION

Dengue is transmitted via Aedes mosquitoes affected by one of the four serotypes of dengue virus (Strauss et al., 2017; Bliman and Coelho, 2016). The general symptoms of dengue include high fever, pain in the eyes, muscle and/or bone. The potentially fatal Dengue Haemorrhagic fever (DHF) is suspected when the patient developed warning signs such as red spots or patches on the skin, bleeding from nose or gums and vomiting blood (Center for Disease Control and Prevention, 2012).

"Predict the Next Pandemic Initiative" was launched on 5th June 2015 in response to this global health challenge. For the first time ever, several departments in the U.S. Federal Government, Peru and Puerto Rico jointly released a series a dengue related data on disease incidence, weather, and the environment to promote data-intensive discoveries (Obama White House, 2015). The competition aims to predict the weekly dengue incidences at two cities in South America – San Juan of Puerto Rico and Iquitos of Peru. DrivenData took over the hosting of the competition under "DengAI: Predicting Disease Spread competition" since 2017 for education purpose (DrivenData, 2018a).

1.1 Problem Statement

Dengue which was first detected mainly in South East Asia during 1940s is now a serious public health concern across the subtropical and temperate regions of Americas, Europe and China due to the change in global climate and international travel. A recent study quoted by World Health Organization (WHO) indicated 3.9 billion people in 128 countries are currently exposed to dengue infection. The three regions of Americas, South-East Asia and Western Pacific registered explosive growth of dengue cases from 1.2 million in 2008 to 3.2 million in 2015. Dengue eradication effort faced setbacks with the disease re-emergence in Japanese after seven decades while the specific Dengue Type 3 (DEN 3) serotype was detected again in the Pacific Island countries after more than a decade (WHO, 2018). The disease remains deadly at 2.5% fatality rate as potent vaccine for the different dengue virus serotypes has yet to be developed. Dengue bears a heavy economic toll on numerous countries with the WHO estimated 500,000 hospital admissions per year is linked to the severe dengue. As children are the most vulnerable population segment, the WHO attributed dengue as the culprit behind the critical illness and death of many children in certain Asian and Latin American nations

(WHO, 2018). While vector control and public health campaigns formed the first line of defence, the health officials worldwide are increasingly turning to dengue forecasting to facilitate the timing of the aforementioned preventive measures along with the issuance of travel advice and medical resource allocation to cope with potential dengue outbreaks.

1.2 Aim of the Research

The main purpose of this study is to forecast Dengue based on nowcasting technique and Google Trends to improve real time surveillance system. Least Absolute Shrinkage Selector Operator (LASSO), a form of penalized regression will be used to forecast the dengue cases of two South American cities, four weeks in advance.

As internet penetration increase, more people are using the Google to search for symptoms and treatment of a Particular disease whenever people around them are infected or experiencing such symptoms (Yang, 2017). Therefore, data-driven modelling (by integrating data from multiple sources especially internet and social media) could deliver near real-time and cost-effective surveillance systems in areas where the efficiency of local disease reporting system remains a challenge (Venkatramanan, 2018; El-Metwally, 2015).

1.3 Research Objectives

Objective 1: To construct Dengue nowcasting models for Iquitos, Peru and San Juan, Puerto Rico using feature engineering.

Objective 2: To forecast the Dengue incidences in the two cities.

Objective 3: To compare the two aforementioned cities in terms of forecast.

1.4 Research Questions / Hypothesis

bjective 1: To construct Dengue nowcasting models for Iquitos, Peru and San Juan, Puerto Rico using feature engineering.

Research question 1: What are the relationship between climatic variables and dengue occurrences in the two cities?

Objective 2: To forecast the Dengue incidences in the two cities.

Research question 2: Can the nowcasting models be used to improve real time surveillance system?

Objective 3: To compare the two aforementioned cities in terms of forecast.
Research question 3: How are the dengue cases between the two cities are different?

1.5 Scope of the Research

Table 1.1: Scope of the Research

Country	City	Temporal range for training data	Temporal range for testing data	Google search languages
Peru	Iquitos	2000-2009	2009 - 2013	Spanish
Puerto Rico	San Juan	1990-2009	2009 - 2013	Spanish, English

Table 1 above entailed the scope of this research. The DrivenData – DengAI competition (DrivenData, 2018a) is a secondary research, using the data from the National Oceanic and Atmospheric Administration, United States (NOAA, 2018a). Google Trends time series data for the two cities will be used as additional data source.

Spanish is the official language of Peru (Country Reports, 2018) while Puerto Rico has both Spanish and English as official languages (World Atlas, 2018). Therefore, this research will only consider Dengue related search terms in English and Spanish. Nevertheless, the temporal range for Google Trends data will be limited to what the research can collect.

The forecast accuracy will be judged based on the Mean absolute error (MAE) formula in Section 3.7: Result Evaluation.

1.6 Significance

According to International Society for Disease Surveillance (2015), public healthcare providers and authorities rely on forecast to decide on their course of action during an outbreak. Epidemiologists would like to know when an outbreak starts, peaks and ends. Incident Commanders, on the other hand, want to know whether the current healthcare system can cope with the severity of an outbreak. Based how the disease trend will depart from

previous season, local officials have to make a series of swift calls such as public announcements and medical resources allocation (by increasing number of beds in the hospital, set up temporary hospital facilities etc) (ISDS, 2015; Lee et al., 2017).

Such warning systems could also be used to form early intervention strategies such as targeted vector control aiming at high risk areas, in order to reduce the severity of an outbreak (Chen et al., 2018). Moreover, such alerts could be issued to travellers to avoid tropical and subtropical regions where Dengue outbreak has occurred or is predicted (New Zealand Government, 2018).

Dengue surveillance systems have long relied on the primary official case reporting which took a while to be aggregated into region / state / country level. Frequent post hoc revision further adds to the lagging effect of this channel (Yang et al., 2017). Traditional epidemiology then tried to provide earlier dengue incidence prediction using the SEIR (Susceptibility, Exposed, Infected, Recovered) models. While these biological process driven models can lead to specific vector control strategies, they require input from various biology experts (Freeze, Erraguntla and Verma, 2018) along with field data collection such as mosquitos count from their habitat (Lee, Chung and Hwang, 2016).

Therefore, this research will seek to utilize feature engineered climatic covariates to forecast the dengue incidences four weeks advance, at city level of Iquitos and San Juan.

1.7 Structure of the Thesis

Chapter 2 covered the literature review of dengue situation in San Juan and Iquitos, general dengue prediction framework, Lasso, nowcasting, Google Trends and the related work on the two cities. Chapter 3 detailed the research methodology. Chapter 4 described the integration of multiple data source of different reporting frequency, explorative data analysis, feature selection and feature engineering. The results of the 4 Lasso models were accounted in Chapter 5. Chapter 6 discussed Google Trends data limitation, time series validation, comparison between the Lasso models of two cities, benchmarking against related works and future recommendations.

CHAPTER 2
LITERATURE REVIEW

A thorough review of the dengue situation at the localities along with the diverse set of dengue forecasting methodology was conducted in the bid to identify suitable model in predicting the diseases occurrence in San Juan, Puerto Rico and Iquitos, Peru.

2.1 Dengue situation in San Juan, Puerto Rico

San Juan is located at the north eastern of Puerto Rico which forms Part of the Hispanic Caribbean Island (Torres et al., 2017). The country first reported dengue fever back in late 1970s (Laureano-Rosario et al., 2018). San Juan reported higher incidences of dengue due to tropical monsoon climate and population density stood at a two and half times much higher than Iquitos (Sougata, Acebedo and Chua, 2017).

Torres et al. (2017) noted that Puerto Rico went through a lengthy dengue epidemic period in 2010 which was dominated by DENV-1 serotype. This was in contrast to the 2007 epidemic with high reporting of DENV-2 and DENV-3. The systematic review conducted by the authors did not support local weather as crucial factor in explaining the changes in the annual case in Puerto Rico.

2.2 Dengue situation in Iquitos, Peru

Iquitos is the capital of the Amazonia area of Loreto, located at the north Peru. It has equatorial climate. The jungle area has seasonal dengue outbreak during the rainy season around March. Torres et al. (2017) stated that DENV-1 and DENV-2 serotypes concentrated in the urban areas with high mosquito density and historical dengue occurrence. The authors also highlighted that more severe dengue cases were reported among those below 15 years old and have a history of dengue infection.

The Andean area, where Peru is geographically situated experienced heavy downpour induced by the La Niña ocean-atmosphere phenomenon in early 2011 which made possible conducive environment for mosquito breeding (The International Federation of Red Cross and Red Crescent Societies, 2011c). Consequently, Loreto was the worst hit area in the country's 2011 most serious dengue crisis (The International Federation of Red Cross and Red Crescent Societies, 2011a). The epidemic was dominated by serotype VD2, which overtook serotype

VD1 and VD4 that were prevalent in December 2010 and previous years respectively (The International Federation of Red Cross and Red Crescent Societies, 2011b).

2.3 General dengue prediction models

There are two main schools when it comes to dengue prediction modelling – mechanistic vs. empirical approaches. Mechanistic models are anchored on the biological stages involved in the vector's life cycle. The SEIR model (S: Susceptibles; E: Exposed; I: Infectious; R: Recovered/Immune) is a common framework used to illustrate the progression of dengue disease (Cheepsattayakorn and Cheepsattayakorn, 2018). Some notable models in this area include Container Inhabiting Mosquito Simulation (CIMSIM), Fock/DENSiM (Dengue Simulation Model) and Dynamic Mosquito Simulation Model (DyMSiM) (Morin et al., 2015).

Some researchers, on the other hand, prefer the "lighter" empirical / statistical approach which focuses more on improving forecast accuracy rather than experts' knowledge of the entire process and parameters. Lee, Chung and Hwang (2016) promoted artificial neural network (ANN) under the empirical school of thought. Specialized time series models such as Auto-Regressive Integrated Moving Average (ARIMA) have also been used to forecast dengue cases. LASSO model overcome the weaknesses of the above mentioned models in terms of the inflexibility to include a myriad of predictors (in mechanistic models) or challenging interpretability (of "black box" machine learning models).

2.4 Least Absolute Selection and Shrinkage Operator (LASSO)

2.4.1 Advantages

LASSO is Part of the penalized regression family along with Ridge and Elastic net. LASSO has the multiple benefits of performing feature selection to produce parsimonious model which prevent over-fitting while overcoming multi-collinearity.

2.4.2 Technical details

The complexity of a regression model increases as more features were introduced in the bid to reduce parameter bias. However the trade-off between error due to bias and error due to variance as the constituents of total error resulted in prediction with low bias but high variation (Figure 2.1). In other word, overfitting was resulted as the model does not generalize well beyond the training dataset (Kane, 2015).

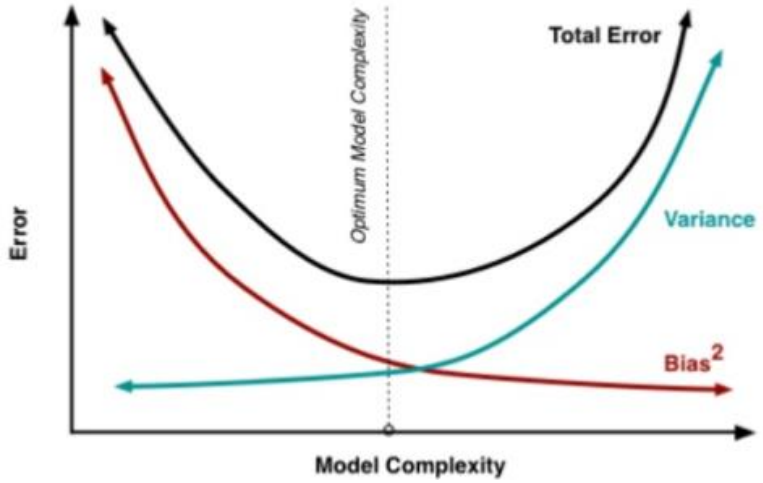


Figure 2.1: Bias and variance trade-off (Kane, 2015)

Thus a small value of Shrinkage Estimator, λ was introduced to "regularize" or shrink the magnitude of the coefficient as Part of the remedy for overfitting. The unbiased OLS estimate was sacrificed in order to achieve parameter estimates with less variance (Kane, 2015).

$$\beta'_k = \frac{1}{1+\lambda}\beta_k \tag{1}$$

where

β'_k = Parameter estimate

β_k = OLS estimate

λ = Shrinkage Estimator and $\lambda \gg 0$

Ridge Regression is a variant of the new regression techniques which deploy λ to handle multi-collinearity among the predictors. However, Ridge does not yield parsimonious models as coefficients are reduced but not zerolized (left hand side of Figure 2.2). Lasso overcomes Ridge's weakness in feature selection by forcing the least important coefficients to zero. The most influential predictor will be identified by Lasso as its coefficient will need the largest λ to be zerolized.

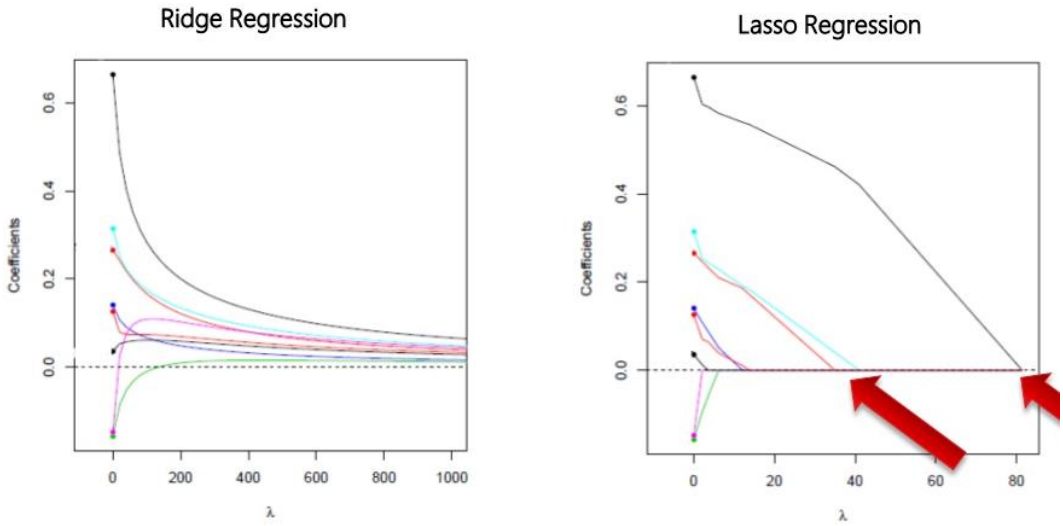


Figure 2.2: Ridge Trace for Ridge Regression vs. Lasso Regression (Kane, 2015).

Lasso $\ell 1$ uses the $\|\beta\|_1$ penalty while Ridge $\ell 2$ uses the $\|\beta\|_2^2$ penalty (Kane, 2015). Lasso characterizes a convex problem (Tibshirani, 2011) and has higher efficiency when it comes to storage and computation (Jain, 2017).

$$\hat{\beta}^{\text{lasso}} = \text{argmin } \|y - X\beta|_2^2 + \lambda \|\beta\|_1$$
$$\hat{\beta}^{\text{ridge}} = \text{argmin } \|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2 \qquad (2)$$

While Robert Tibshirani introduced Lasso back in 1996, it was popularized only in 2002 when Least-Angle Regression (LARS) algorithm modified forward stagewise regression to derive an effective Lasso solution (Tibshirani, 2011).

Lasso models could have lower Mean Square Error (MSE) in comparison to Ridge as influential predictors which are correlated with the chosen predictors are zerolized by Lasso. Elastic Net, a new Ridge-Lasso hybrid tries to balance between the 2 models by allowing grouped selection of correlated features (Kane, 2015; Jain, 2017).

$$\hat{\beta}^{\text{ridge}} = \text{argmin } \|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2$$

$$\hat{\beta}^{\text{lasso}} = \text{argmin } \|y - X\beta|_2^2 + \lambda \|\beta\|_1$$

$$\hat{\beta}^{\text{elastic}} = \text{argmin } \|y - X\beta\|_2^2 + \lambda_2\|\beta\|^2 + \lambda_1\|\beta\|_1 \quad (3)$$

2.5 Nowcasting with LASSO

Nowcasting is the methodology of using currently available predictors' data to extrapolate occurrence of an interested outcome in near future. Nowcasting which was originated in meteorology (World Meteorological Organization, 2017) is now being adopted in other fields such as finance to understand the current state of economy before the release of much delayed, official macro economy indicators from central banks (European Central Bank, 2013; CRAN, 2018). Medical prognostic studies such as Musuro et al. (2014) used Lasso to forecast Chronic Respiratory Questionnaire (CRQ) dyspnea for 6 months in advance.

According to Shi et al. (2016), correlative statistical approaches are superior to the conventional population dynamic models when it comes to nowcast dengue. The former group allows for integration with real time input data and tend to be accurate if forthcoming circumstances do not deviate from the historical pattern. The authors informed that statistical dengue nowcasting attempts since 2011 include Autoregressive Integrated Moving Average (ARIMA), Knorr-Held two-component (K-H), Poisson multivariate regression, step-down linear regression, generalized boosted regression, and negative binomial regression. Recent researches look into LASSO as nowcasting model for its feature selection benefit along with the capability of incorporating autoregressive terms as predictors just as in the classical time series models (Chen et al., 2018). The need for cross validation in finding the optimal penalty is a step not found in the aforementioned statistical models. The final LASSO model is geared for real life implementation as it was calibrated with multiple training and validation stages (Shi et al., 2016).

The LASSO-based dengue nowcasting framework operated by Singapore's National Environment Agency (NEA) was documented by Shi et al. (2016). Multiple data streams were integrated on weekly basis – dengue cases, Breeding Percentage (BP, the dominance of Ae. Aegypti among the breeding sites), weather (temperature and humidity), population from

midyear census, seasonality and trend decomposition by Breaks For Additive Seasonal and Trend (BFAST) algorithm in R. A total of 226 covariates (with lags up to 20 weeks) were used to construct 12 LASSO sub-models corresponding to forecast window of 1-12 weeks ahead in a comprehensive training-validation-testing cross validation. A final ensemble model was built to derive an overall prediction for next 12 weeks. Higher average weekly temperatures and BP were significant predictors over the short term while high levels of absolute humidity over the last month influenced medium term prediction. The LASSO ensemble outperformed Seasonal AutoRegressive Integrated Moving Average (SARIMA) and step-down linear regression in terms of slowest decay rate of MAPE and high percentage of actual cases falling within the 95% confidence interval. While the outbreak size prediction of the LASSO ensemble underperformed its epidemic timing prediction, it withstood the test of reliability whereby advance alerts were raised during the 2013 and 2014 dengue epidemic. The Singapore Ministry of Health and the Environmental Public Health Operations Department of the NEA were able to redirect hospital sources and dengue campaign accordingly.

Chen et al. (2018) evaluated LASSO algorithm in nowcasting infectious diseases in Japan (chickenpox and HFMD), Singapore (dengue and HFMD), Taiwan (dengue) and Thailand (chickenpox, dengue and malaria). Wavelet analyses was first performed to study the periodicity (amplitude, time and frequency) of each disease in each locality, out of which lags of disease incidence and climatic variables up to 26 weeks were determined as autogressive predictors. The dengue incidences underwent logarithmic transformation while the predictors where standardized prior to modelling. Multiple LASSO models were created for each disease in different countries for 1-26 weeks ahead. The optimal constraint parameters were determined using ten-fold cross validation in R. Additional simplistic incidence only models were built to evaluate the value added of building complex models using historical dengue cases (linear and squared post transformation), historical climatic variables (temperature and relative humidity) and calendar month indicators. The ability of models to act as early warning system was tested by identifying the threshold corresponding to previous year's 75^{th} percentile high incidence, alongside MAPE, RMSE and R-squared. While high accuracy was reported for HFMD in all countries, the performance of the other models waned beyond 4 weeks horizon. The climatory variables were found to add more value in temperate region where cyclic patterns were observed for the diseases and climatic data, in comparison to the more stable climate in equatorial region. The calendar month indicator was found to be

influential in certain models, such as Thailand's chickenpox. The models are better than predicting the timing, rather than the magnitude of disease outbreak.

The predictive prowess of LASSO can oftentimes be boosted using crowd generated signals from the internet and social media such as the Google Trends.

2.6 Google Trends

Google Trends (https://trends.google.com/trends/) was launched in 2006 while the underlying data dates back to 2004 (Seung-Pyo, Hyoung and San, 2018). Google compiled the internet searches from its three platforms, viz. Search, Google News, or YouTube using its in house Knowledge Graph technology (Google News Initiative, 2018a). Google Trends (GTs) analyses the proportion of topical searches conduction over a time range from various localities and languages. The public accessible version of GTs is normalized on a 0–100 scale with 100 being the maximum value of the time frame selected. The value of zero was used to indicate searches that below a certain volume threshold instead of absence of search. A downward trend is to be interpreted as the reducing popularity of search term relative to other topics (Google News Initiative, 2018b).

GTs sparked great interest as a real time explanatory variable in epidemiology research on the premise that the more a person is affected by a disease, the more likely that he or she will query Google for related information (Yang et al, 2017). Google hosted the Google Flu Trends (GFT) and Google Dengue Trends (GDT) for limited number of countries from 2008 until 2014 (Google, 2018). Both Peru and Puerto Rico where DengAI competition are not in GDT list (Google, 2018).

Yang et al. (2017) cited a research which reported inconsistent performance of GDT across Mexico states. However, earliest publications in 2011 demonstrated that Google Trend data can be used provide timely prediction of dengue cases in Bolivia, Brazil, India, Indonesia and Singapore (Seung-Pyo, Hyoung and San, 2018).

An influenza focused ARGO (AutoRegressive model with GOogle search queries as exogenous variables) model was extended to predict national level dengue cases in Brazil, Mexico, Singapore, Taiwan and Thailand (Yang et al, 2017). Autogressive models were built for each country using lags of 1-12 and 24 months under a rolling two year window. Next, top

ten dengue-related search terms for respectively country were identified and incorporated into the earlier autoregressive models to form ARGO. Lasso, the L1 regularization was tweaked to prevent the coefficient of important time lags from being zerolized. Four alternative models were constructed – a seasonal autoregressive (SAR) model; two models using official Google Dengue Trends (GDT) and GTs as explanatory variables respectively; a combination of SAR and GDT; naïve model with 1 month lag autoregressive term. ARGO consistently outperformed the rivals by RMSE, MAE, RMSPE, MAPE and Pearson correlation standards, except in Taiwan. The authors attributed the outlying Taiwan model performance to the country having minimal dengue occurrence until the 2014-2015 outbreak plus Google is not the leading search engine in Taiwan. Further tests revealed that ARGO is robust to both the sampling variation of GTs (by collecting the search terms in ten different sessions) and the scenario that reporting delay caused the latest month dengue cases to be unavailable as autoregressive term. Consequently, the authors concluded that ARGO framework is flexible to forecast dengue in multiple countries with high Google market share and seasonal dengue outbreak patterns.

Teng et al. (2017) appraised Google Trends (GTs) as a source of real time, external regressor to complement time series model in predicting Zika virus disease (ZVD). The global ZVD case data from February until November 2016 were downloaded from the websites of Pan American Health Organization (PAHO) and World Health Organization (WHO). The "Interest over time" measure of Zika related search terms during the same window was downloaded from GTs website and was normalized to be 100 on the 1st day of the training dataset. The data compiled was segregated into two datasets, i.e. training (the initial 37 weeks) and testing (final 3 weeks which corresponds to Zika epidemic). Pearson Product-Moment Correlation during the training period established that strong linear correlation exist between Zika related GTs and reported ZVD cases. A simple linear regression was constructed as the first baseline model. A second baseline model of ARIMA (0, 1, 3) was fitted using Box-Jenkins method. An augmented ARIMA (0, 1, 3) was constructed using GTs as additional explanatory variable. The augmented ARIMA model registered lower Akaike Information Criterion (AIC) than the linear baseline model and the basic ARIMA model.

2.7 Related work on San Juan and Iquitos from the competition

2.7.1 Non-ensemble models

Laureano-Rosario et al. (2018) made use of the climatic and population variables from NOOA to forecast the weekly dengue incident rate (per 100,000) for two population subgroups. Confirmed daily cases specific to the population at risk (those younger than 24 years old) and vulnerable population (small children and old folks) were obtained from two medical sources. Sea Surface Temperature (SST) data were extracted from NOAA Advanced Very High Resolution Radiometer (AVHRR) as an additional predictor. The authors ran two multiple linear regression as baseline models. The ANN from Radar Pluvial flooding Identification for Drainage System (RAPIDS) was modified with factor, a = 3 and genetic algorithm II (NSGA-II) to create two ANN models for the two population subgroups. The ANNs were run as binary models on whether there would be a potential dengue outbreak against predetermined thresholds for the each population subgroups. The authors found the non-linear ANN outperformed the MLR as listed in Table 2. However, ANN models have lower accuracy rate for the "potential outbreak" period in comparison to the "no outbreak" period. Population size and date were found to have an excitatory influence (positive ANN weights) upon dengue incidents while previous cases was found to be an inhibitory influence (negative ANN weights). Maximum air temperature seemed to have inconsistent impact on the two population subgroups.

Table 2.1: The result of ANN vs. Multiple linear regression conducted by Laureano-Rosario et al. (2018)

Predict weekly dengue incident rates	Method	Performance metrics	Significant predictors (ANN weight, MLR coefficient)	
			Positive influence	Negative influence
Population at risk (younger than 24 years old)	ANN	AUC = 0.91 FM = 0.97	• Population size • Date • Maximum air temperature	• Previous cases
	MLR	r2 = 0.74	• Previous cases	• Date • Population size
Vulnerable population (younger than 5 years and older than 65 years)	ANN	AUC = 0.71 FM = 0.81	• Population size • Date	• Previous cases • Maximum air temperature
	MLR	r2 = 0.33	• Minimum air temperature	• Date • Population size

			• Previous cases	

Freeze et al. (2018) expanded their Data Integration and Predictive Analysis System (IPAS) for Influenza like Illness (ILI) to predict dengue cases in San Juan and Iquitos. Feature engineering was mostly centred on the weekly dengue incidences with normalization of dengue incidence to per hundred thousand of annual population; square and cube of the normalized dengue incidences as nonlinear terms; slope or the change in normalized incidence over 1-4 week horizons for trend analysis. R's caret package was used to create combined models to predict dengue incidences for 1-week and 4-weeks ahead. Features were filtered based on the significance of their coefficients in a Multiple Linear Regression (MLR) model. Shortlisted features were used in the final MLR, Support Vector Machine (SVM), Random Forests (RF) and Boosting models. Significant features found comprised of seasonal effect (represented by week number), nonlinear terms, trends and regional effect (San Juan vs. Iquitos). Random Forest and Boosting had the smallest and largest testing Mean Square Error (MSE) respectively.

2.7.2 Ensemble Models

Yamana et al. (2016) highlighted the need to reconcile the competing disease forecasting models. The authors created three sub-models (F1, F2 and F3) without using the climatic and population data provided in the competition. F1 involved making simplistic assumptions about the distributions of S, I, D, R0 components within the Susceptible–Infectious–Recovered (SIR) model. Next, multiple SIR models was aggregated using Ensemble Adjustment Kalman Filter (EAKF) data-assimilation method. It under-predicted pre-dengue outbreak incidences. F2 enlisted Bayesian Model Averaging (BMA) in forecasting current week's dengue incidence as the weighted sum of the previous weeks' dengue incidences from the previous seasons. It was the best peak incidences timing predictor among the three models. F3 focused on the historical likelihood whereby peak timing was proxy by Gaussian distribution while Gamma distribution was used to generate peak incidence and total incidence in a season. Its forecast deteriorated for outbreaks which did not converge to long term average. Superensemble (SE) models consisting of different combinations of F1, F2 and F3 were constructed by weighting their respective performance in the preceding weeks using BMA. SEs reported overall lower Mean Absolute Error (MSE) than the individual models, implying the strength of SE framework in offsetting the bias within sub-models. The

reliability of F2 among the individual models was carried forward into the outperformance of SE(F1, F2) among the SEs.

The team of Johnson et al. (2018) was one of the six winners of the 2015 Dengue Forecasting Challenge project. The weekly incidences were first of all, underwent a novel square-logarithmic transformation before followed by the creation of hetGP and GLM. hetGP (heteroskedistic Gaussian Process) which disregard climatic variables in favour of a phenomenological approach. It overcomes heteroscedasticity by deploying a nonparametric Gaussian Process (GP) regression fitting. The model relies on four predictors (season-time, sine wave, starting level and severity indicator) to match current season to the most similar historical trajectory. Generalized Linear Model (GLM) involved egative binomial with a log link while augmenting given climatic variables (excluding humidity) with Southern Oscillation Index (SOI) and value of El Nino 1/2. The covariates underwent extensive feature engineering which entailed the combination of autoregressive (climatic lags), trend (cumulative, average, smoothing, simple linear regression), trigonometric (sine and cosine functions to capture seasonality) and transformation (logarithmic and squared) elements. Monte Carlo simulations were run to generate the forecasts followed by evaluation against the other top entries from the competitions plus Yamana et al. (2016). The hybrid hetGP was among the top three models for San Juan while its performance was less consistent for Iquitos. The authors noticed that the parsimonious F2 model from Yamana et al. (2016) shared the advantageous historical seasonal pattern memorization and matching capability as their hetGP.

Sougata, Acebedo and Chua (2017) performed massive feature engineering with the creation of 103 new variables from Mosquito lifecycle related lagged effects, up to 32 weeks, interaction between climatic variables and forecasted dengue cases from decomposition-based time series models. An ensemble model took the median prediction from the following sub-models, namely MLR, WMLR and CPM. Multiple Linear Regression (MLR) used overall data which enabled San Juan and Iquitos to learn from each other's dengue experience. Weighted Multiple Linear Regression (WMLR) emphasized more on the epidemic periods. Covariate Pattern-Matching (CPM) relied on the closest matching historical window of each 19 raw climatic variables to predict the daily case changes. CPM is likely to overfit the training data as its testing dataset's MAD of 26.05% was much larger than training set's (at 7.23%). The ensemble model outperformed all sub-models, with the exception for CPM in

the training set. The authors were able to deduce from the significant coefficients from MLR and WMLR models that vegetation and precipitation had negative impact while humidity and maximum temperature had positive influence on dengue occurrence.

Buczak et al. (2018) was another winning team from the 2015 challenge. Their ensemble algorithm weighted the top 300 performing models from among the following by comparing their forecast errors in the past four years: (1) 1248 distinct additive seasonal Holt-Winters (HW) models which were created by tuning various parameters (with / without wavelet smoothing, periods of seasonality, ending weeks, optimization using RMSE or MARE). HW pretty much dominated the ensemble. (2) Two-dimensional Method of Analogues models which relied on historical sequence of precipitation with 2-weeks lag to forecast the weekly dengue incidences for 4-52 weeks in advance. These models performed better for San Juan's peak height and total cases. (3) Simple historical models of peak height, peak weak and total cases. They only outperformed during the start of individual forecast years, thus are the weakest of the three sub-models. The authors' team was the strongest contender for Iquitos' seasonal peak height and total cases but their San Juan predictions were mixed.

2.7.3 Gaps identified within literature review

According to National Oceanic and Atmospheric Administration (2018a), with the exception of dengue in the study locations or nearby locations, participants are allowed to augment the model with exogenous data sources, such as social media and demography. However, most participants mostly stick to the climatic data, especially to take advantage of the satellite reanalysis data which is not found in most dengue forecasting research.

Also, most of the Participants approached the competition purely from historical perspective. This is evidenced by only one of the six papers published (Freeze, Erraguntla and Verma, 2018) which truly adopted forward looking nowcasting (forecasting the present or into the near future) which is most practical for situational awareness (Marques-Toledo et al., 2017). Buczak et al. (2018) adopted nowcasting in one of the sub-model but not at the ensemble model level.

LASSO has been used in dengue nowcasting but none of them utilized internet searches as exogenous regressor (Shi et al., 2016; Chen et al., 2018). The limited dengue forecasting model which incorporated GTs on the other hand, was not built for nowcasting and did not

take climatory conditions into consideration (Yang et al, 2017). Hence, GTs can be utilized as external regressor to supplement the official DengAI climatic dataset to nowcast dengue cases in Iquitos, Peru and San Juan, Puerto Rico under a LASSO framework.

2.7.4 Summary

Table 2.2: Significant predictors and weakness of related work

Citation	Method(s)	Significant predictors	Strength	Weakness
Freeze et al. (2018)	Separate o MLR (with non-linear terms) o SVM o RF o Boosting	o Week number (seasonal effect) o Total number of cases o Square and cube of the total cases (nonlinear) o Previous weeks' incidences and the percent change in incidences from previous weeks (trend) o Temperature and daily temperature range (climate) o Region	NA	All models had trouble estimating the severe outbreak for Iquitos in 2010
Yamana et al. (2016)	F1N - Susceptible–infectious–recovered ensemble adjustment Kalman filter		Accuracy in peak timing & incident tend to improve after the crystallisation of season peak	Tend to predicted false, very small peaks in the beginning of a low dengue case season
	F2N - Bayesian weighted outbreaks	o	More accurate peak timing prediction than F1 and F3	The more deviation from historical pattern, the larger the errors
	F3N - Historical likelihood	o	Slow adaptation to outbreaks that deviate from the long-term average	Less prone to huge error

	Super ensembles (F1N + F2N + F3N)	NA	Lower MAD for individual models	
Johnson et al. (2017)	Hybrid of hetGP and GLM	NA	NA	NA
Sougata, Acebedo and Chua (2017).	Multiple Linear Regression (MLR) on combined data	<u>Positive</u> o Max temperature (current) <u>Negative</u> o Vegetation (lagged) o Precipitation (lagged)	NA	NA
	Weighted Multiple Linear Regression (WMLR)	<u>San Juan – Positive</u> o Max temperature (current, lagged) <u>San Juan – Negative</u> o Vegetation (lagged) o Precipitation (lagged) <u>Iquitos - Positive</u> o Max temperature (lagged) o Rel. Humidity (lagged) <u>Iquitos - Negative</u> o Avg. Temp (lagged)	NA	NA
	Covariate Pattern-Matching	NA	Established Linkage between mosquito lifecycle and empirical model.	CPM is likely to overfit the training data as its testing dataset's Mean Absolute Deviation (MAD of 7.23%) was much larger (at 26.05%).
	Ensemble (median of 3)	NA	Better performance than individual models	NA
Buczak et al. (2018)	Two-dimensional Method of			

	Analogues models			
	Additive seasonal Holt-Winters models - with and without wavelet smoothing		Dominated the ensemble	Very sensitive to different periods of seasonality and to training end points
	Simple historical models			Tend to perform best at the beginning of the forecast year but with a subsequent decrease in performance so that they often were not included in later forecasts.
	Ensemble		Scored higher for peak height and total dengue case counts reported in a transmission season for Iquitos than all other models submitted to the Dengue Challenge.	San Juan - accuracy of predictions for the total number of seasonal dengue cases and the peak weekly number of dengue cases, predictions were mixed

Based on Table 2.2 above, the same thread of finding corresponding historical climatic pattern to predict dengue can be traced in Yamana et al. (2016)'s F2 BMA, Johnson et al. (2018)'s hetGP and Sougata, Acebedo and Chua (2017)'s CPM models. The timing of Iquitos' peak heights remained a challenge that all modellers failed to handle them well.

Ensemble models are the favourite trend among researchers as higher accuracy was achieved by diverse models complementing each other's weaknesses. However, the models' interpretability was sacrificed as only Sougata et al. (2017) managed to relate the results of their simpler MLR and WMLR sub-models to the climatic impact on mosquito lifecycle. Furthermore, depending on the complexity of underlying component models, most public healthcare providers may lack the expertise and resources to maintain the ensemble models for practical surveillance. Therefore, striking a balance between modelling accuracy and being able to notify public health decision makers on timely basis are crucial in effective dengue surveillance (Freeze, Erraguntla and Verma, 2018).

CHAPTER 3
RESEARCH METHODOLOGY

3.1 Data Collection

This research will involve three data sources from DrivenData, National Oceanic and Atmospheric Administration (NOAA) and Google Health API. This is because the competition permits the usage of other publically available data sources such as social media or demographic data (NOAA, 2018b). Johnson et al. (2017), one of the 2015 winning made extensive used of El Nino related weather variables which was not provided by NOAA then.

The current competition host, DrivenData (2018b) is the primary data source. It provides three csv files (dengue_features_train.csv, dengue_labels_train.csv and dengue_features_test.csv) which contain the climatic variables. The metadata for the raw files are given in Table 4. It eased data aggregating job by standardizing the inconsistent reporting frequencies (daily and weekly) in the original NOAA datasets into weekly.

Table 3.1: Metadata for Climatic Variables

Climatic Data	Field	Details
City and date indicators	city	City abbreviations: sj for San Juan, iq for Iquitos
	week_start_date	Date given in yyyy-mm-dd format
NOAA's GHCN daily climate data weather station measurements	station_max_temp_c	Maximum temperature
	station_min_temp_c	Minimum temperature
	station_avg_temp_c	Average temperature
	station_precip_mm	Total precipitation
	station_diur_temp_rng_c	Diurnal temperature range
PERSIANN satellite precipitation measurements (0.25x0.25 degree scale)	precipitation_amt_mm	Total precipitation
NOAA's NCEP Climate Forecast System Reanalysis measurements (0.5x0.5 degree scale)	reanalysis_sat_precip_amt_mm	Total precipitation
	reanalysis_dew_point_temp_k	Mean dew point temperature
	reanalysis_relative_humidity_percent	Mean relative humidity
	reanalysis_specific_humidity_g_per_kg	Mean specific humidity
	reanalysis_precip_amt_kg_per_m2	Total precipitation
	reanalysis_max_air_temp_k	Maximum air temperature

	reanalysis_min_air_temp_k	Minimum air temperature
	reanalysis_air_temp_k	Mean air temperature
	reanalysis_avg_temp_k	Average air temperature
	reanalysis_tdtr_k	Diurnal temperature range
Satellite vegetation - Normalized difference vegetation index (NDVI) - NOAA's CDR Normalized Difference Vegetation Index (0.5x0.5 degree scale) measurements	ndvi_se	Pixel southeast of city centroid
	ndvi_sw	Pixel southwest of city centroid
	ndvi_ne	Pixel northeast of city centroid
	ndvi_nw	Pixel northwest of city centroid

The weekly frequency of dengue related search terms to be extracted from the third data source, i.e. Google Trends. This research obtained the approval from The Google Trends team to use their pythonic Google Health API in October 2018. Since both Peru and Puerto Rico are mostly Spanish speaking nations, the Google Trends data collected will be done using two languages, i.e. English and Spanish. The list of query terms used by Yang et al. (2017) to analyse Mexico and Singapore will be adopted for Spanish and English languages respectively as per listed in Table 3.2.

Table 3.2: Query terms to be extracted from Google Trend API from the two cities

Spanish (from Mexico)	English (from Singapore)
Dengue dengue.dengue.dengue el.dengue dengue.sintomas sintomas.del.dengue dengue.hemorragico sintomas.de.dengue que.es.dengue dengue.clasico dengue.mosquito	dengue dengue.fever dengue.symptoms symptoms.dengue.fever symptoms.of.dengue dengue.mosquito mosquito

3.2 Preprocessing of population dataset

The primary dataset is reported on weekly basis while the aforementioned annual population data from NOOA is on annual basis. Therefore, the reporting frequency needs to be standardized. The weekly population of each city will be calculated separately using upsampling (Brownlee, J., 2016) and interpolation techniques (The SciPy community, 2018; The pandas project, 2018a). Linear interpolation by Johnson et al. (2017) will be the default method upon graphical inspection of the fitted data using logistic model (Khan Academy, 2018). This will also solve the missing population data of San Juan from 1991 – 1998, rather than carrying forward the 1990 population data for the missing 8 years as per Freeze et al. (2018). A combined (row binding) population file will be created by merging the two population files with a new column for city label of sj and iq.

3.3 Preprocessing of Google Trends dataset

The weekly frequency of all the dengue related search terms extracted in English and Spanish will be summated for the week. An overall Google Trend file will be created (row binding) with the city label of sj and iq.

3.4 Preprocessing and Explorative Data Analysis (EDA) of combined dataset

First of all, a new variable named "flag" will be created in the DrivenData's csv files before merging them to create the primary dataset as per Table 3.3.

Table 3.3: Flags to merge and break datasets

Original file	Flag	Binding method
dengue_features_train.csv	Training	
dengue_features_test.csv	Test	Row wise
dengue_labels_train.csv	Training	Column wise, inner join

Secondly, the preprocessed datasets from NOAA and Google Trends will be inner-joined (column binding) with the primary dataset using the common indices of city (sj / iq) and week_start_date. Thirdly, the variables will undergo rescaling exercise in Table 3.4 to standardize the measurement units.

Table 3.4: Rescaling of variables

Variable	Transformation	Note
Total_case	• Normalization by dividing the weekly case by the interpolated weekly population, then multiply by 100,000. • Follow by log (1 + normalized case).	• Laureano-Rosario et al. (2018) and Freeze et al. (2018) used the raw annual population data throughout the year. • Laureano-Rosario et al. (2018), Chen et al. (2018) and Shi et al. (2016) applied logarithmic transformation. • Johnson et al. (2017) utilized a combination of square root and logarithmic functions.
• reanalysis_dew_point_temp_k • reanalysis_max_air_temp_k • reanalysis_min_air_temp_k • reanalysis_avg_temp_k • reanalysis_tdtr_k • reanalysis_air_temp_k	Minus 273.15 to convert from Kelvin to Celsius.	This was practiced by Sougata, Acebedo and Chua (2017).

Fourthly, the univariate analysis of all variables will be performed. Descriptive statistics (The pandas project, 2018c) and the distribution of all variables by cities will be compared using the violin plot and shaded kde plot (Waskom, 2018a; Koehrsen, 2018). According to Koehrsen (2018), Kernel Density Estimation (KDE) plots are preferred over the traditional histogram for the improved readability for multiple categories comparison.

Outlier assessment using subplots of boxplots by cities will follow suit. Only Sougata, Acebedo and Chua (2017) reported on outliers whereby they are not treated as no outlandish value was detected. Next, the overall pattern of missing values will be inspected using pandas-profiling (Akinfaderin, 2017; Yilmaz, 2017; Python Software Foundation, 2018).

Fifthly, bivariate analysis in the form of correlation analysis by city will take the next centre stage. Pairwise Pearson coefficient (The pandas project, 2018b) will be evaluated by via annotated heatmap (Waskom, 2018b) for all variables. The correlation between each city's total _case vs. independent variables will be further zoomed in using bar chart in descending order. The relationship amongst the similar climate variables across different measurement systems as listed in Table 3.5 will also be analysed to check for potential dimension reduction opportunity.

Table 3.5: Potential multi-collinearity amongst similar climatic variables

Measurement systems	NOAA's GHCN daily climate data weather station	PERSIANN satellite precipitation	NOAA's NCEP Climate Forecast System Reanalysis	NOAA's CDR Normalized Difference Vegetation Index
Temperature	• station_max_temp_c • station_min_temp_c • station_avg_temp_c • station_diur_temp_rng_c		• reanalysis_dew_point_temp_k • reanalysis_max_air_temp_k • reanalysis_min_air_temp_k • reanalysis_avg_temp_k • reanalysis_tdtr_k • reanalysis_air_temp_k	
Precipitation	• station_precip_mm •	• precipitation_amt_mm	• reanalysis_precip_amt_kg_per_m2 • reanalysis_sat_precip_amt_mm	
Normalized difference vegetation index (NDVI)				• ndvi_se • ndvi_sw • ndvi_ne • ndvi_nw

Sixthly, please refer to Table 3.6 whereby the treatment of missing values will then be determined based on the amount and type of missing data plus the correlation with other variables. After ensuring that is no more missing values, all independent variables will be standardized using mean and standard deviation (scikit-learn developers, 2018a) as Part of the prerequisite of LASSO model (Chen et al., 2018; The Pennsylvania State University, 2018).

Table 3.6: Proposed missing variable treatment strategy

Scenario	Remedy
A variable with more than 50% missing data	Exclude the variable from analysis
A variable with missing variable can be proxied by other variables	The variable could be dropped to reduce multi-collinearity
Missing completely at random (MCAR)	Mean, median, mode and K-nearest neighbour (KNN) imputations as suggested by Akinfaderin(2017) . Or backfill, frontfill or nearest neighbour are plausible

	imputation methods (Stack Exchange Inc., 2016b).
Missing at Random (MAR)	Multiple Imputation using MICE (Multiple Imputation by Chained Equations) (Akinfaderin, 2017)
Missing not at Random (MNAR)	Create distinct "missing" group since there is not enough information in the observation can be used to impute the missing values (ResearchGate, 2018; Das, 2016).
San Juan's missing 1991 – 1998 annual population	Interpolation (Johnson et al., 2017; Khan Academy, 2018)

Seventhly, the standardized variables will be examined for low variance threshold (Stack Exchange Inc, 2016c; Stack Exchange Inc, 2016d). Features with low information will be filtered using p = 0.8 (scikit-learn developers, 2018g) in order to reduce the number of lagged climatic variables to be created in the next stage. p = 0.8 means 80% of the values of a variable are of the same value.

Eighthly, the lifecycle of mosquito and the incubation period of DENV (World Health Organization, 2018) could cause delayed in the impact of climatic variables on total_cases. Autocorrelation function (ACF) will be deployed to inspect the correlation between historical and the latest log transformed total_case (Brownlee, 2017; DataCamp, 2018). The corresponding significant lagged log transformed total_case identified from ACF will be created as autoregressive predictors as adopted by Laureano-Rosario et al. (2018) and Chen et al. (2018). Different lagged climatic variables were found to be significant in San Juan and Iquitos, ranging from the minimum 1 week by Laureano-Rosario et al. (2018) up till maximum 32 weeks by Sougata, Acebedo and Chua (2017). Therefore, lagged climatic variables will be derived up to 32 weeks using loop and shift function (Augspurger, 2016).

3.5 Training, validating and testing of LASSO models

The preprocessed training dataset will be further split into training and validation datasets at 70:30 ratio (scikit-learn developers, 2018b). Please refer to table 10 whereby four LASSO models will be constructed using LARSCV with 10-fold cross validation and grid search on the training data (Azencott, 2015; scikit-learn developers, 2018c; scikit-learn developers, 2018d; scikit-learn developers, 2018e). Scoring parameter within the GridSearchCV function will be set to 'neg_mean_absolute_error' in order to minimize MAE (scikit-learn developers, 2018f). Both Freeze et al. (2018) and Chen et al. (2018) reported using 10-fold cross validation for their models.

Table 3.7: Four LASSO models to be constructed

Model	City	Independent variables
M1	San Juan	No lagged climatic variables
M2	San Juan	With lagged climatic variables
M3	Iquitos	No lagged climatic variables
M4	Iquitos	With lagged climatic variables

Numerous lagged climatic variables will result in high dimensionality dataset. LASSO will perform feature selection by zerolising the non-relevant variables (scikit-learn developers, 2018g). This is the last leg of the feature selection strategy as listed in Table 3.8.

Table 3.8: Feature selection / Dimensionality reduction strategy

Stages	Filter criteria
Univariate analysis	• Drop variables with more than 50% missing values. • Features with low information (p = 0.8) post standardization.
Bivariate analysis	Similar climatic variables with high pairwise correlation coefficient
Training of LASSO models	Variables with non-zero coefficient

3.6 Result Evaluation

Performance assessment will be conducted visually and quantitatively.

The finalized models will be applied throughout the entire periods and the log transformed predictions will be reversed (antilog, divide by 100,000 then multiply by the monthly population and finally minus 1) to compare the actual vs. predicted total_cases. This is crucial to examine the ability of the models in capturing the timing and magnitude of the outbreak in testing dataset (Sougata, Acebedo and Chua, 2017).

- The residuals (actual minus prediction) will be plotted against the actual values to check for residual correlations, esp. heterosckedascity in residuals (Freeze, Erraguntla and Verma, 2018).

- The accuracy of the models will be evaluated based on the criteria set by the organizer (NOAA, 2018a), i.e. using Mean Absolute Error (MAE). MAE is the mean absolute difference between predictions $\hat{y}$ and observations y over n data points:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i| \quad (4)$$

The "Regression Coefficients Progression for Lasso Paths" of each finalized model will be plotted (TapVenture, 2017). The list of coefficients from each model will be displayed using bar chart to check the ranking of Google Trends' coefficients. The value added from Google Trend will be determined by whether Model3 and Model4 have lower MAE than Model1 and Model2.

3.7 Benchmarking

The research will be benchmarked against two similar Dengue nowcasting models by:

- Freeze, Erraguntla and Verma (2018) which forecast 1-week and 4-weeks ahead, based on the same DrivenData – DengAI competition.
- Shi et al. (2016) which forecast week 1 to week 12 ahead for Singapore.

3.8 Ethical and Safety Issues

All data from DrivenData (2018a) and Google Trends is of secondary nature. No individual data about Dengue patient or Google user's IP address will be used. The related data has been aggregated to weekly and city level.

CHAPTER 4
IMPLEMENTATION

4.1 Preprocessing of population dataset

Please refer to "PART 1- Preprocess annual population data" of attached Jupyter notebook.

4.1.1 Part 1B - San Juan

Raw 'San_Juan_Population_Data.csv' was imported to plot Figure 4.1A. There are missing data from 1991-1998 and an outlier in year 1999. APart from that, population seems to be on decreasing trend after 2005. The conventional logistic population model (Khan Academy, 2018) is unsuitable to interpolate the missing data given the triangular shape in Figure 4.1.

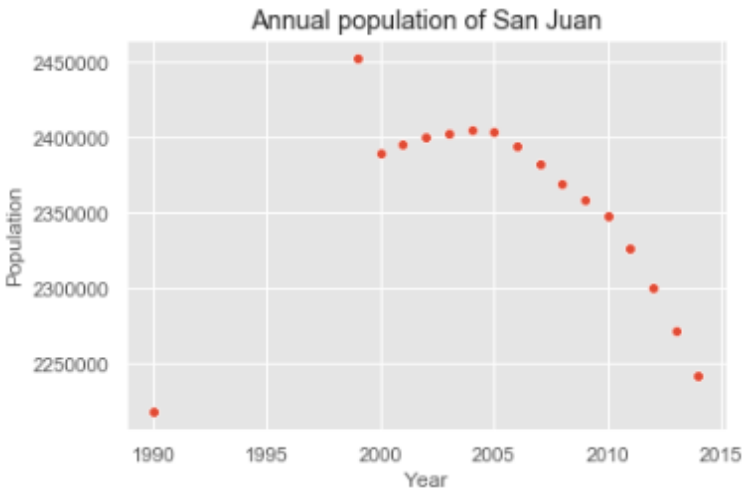


Figure 4.1A: Annual population of San Juan

Figure 4.1B showed the outlier in 1999 was deleted and replaced with np.NaN along with 1991-1998 so that they can be interpolated using "Freq: A-DEC" of time series object, assuming that the annual population data was carried forward from 31st December of previous year.

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 25 entries, 1989-12-31 to 2013-12-31
Freq: A-DEC
Data columns (total 1 columns):
Population    16 non-null float64
dtypes: float64(1)
memory usage: 400.0 bytes
```

Date	Population
1989-12-31	2217968.0
1990-12-31	NaN
1991-12-31	NaN
1992-12-31	NaN
1993-12-31	NaN
1994-12-31	NaN
1995-12-31	NaN
1996-12-31	NaN
1997-12-31	NaN
1998-12-31	NaN
1999-12-31	2389397.0
2000-12-31	2395941.0
2001-12-31	2400500.0
2002-12-31	2403420.0
2003-12-31	2405376.0
2004-12-31	2403542.0
2005-12-31	2394920.0
2006-12-31	2382377.0
2007-12-31	2369802.0
2008-12-31	2358308.0
2009-12-31	2347833.0
2010-12-31	2327325.0
2011-12-31	2300694.0
2012-12-31	2271796.0
2013-12-31	2242285.0

Figure 4.1B: NaN for 1991-1999 (proxy by previous year end)

Resampling method which involved piecewise linear interpolation was used to increase the frequency from annual to daily in Figure 4.1C.

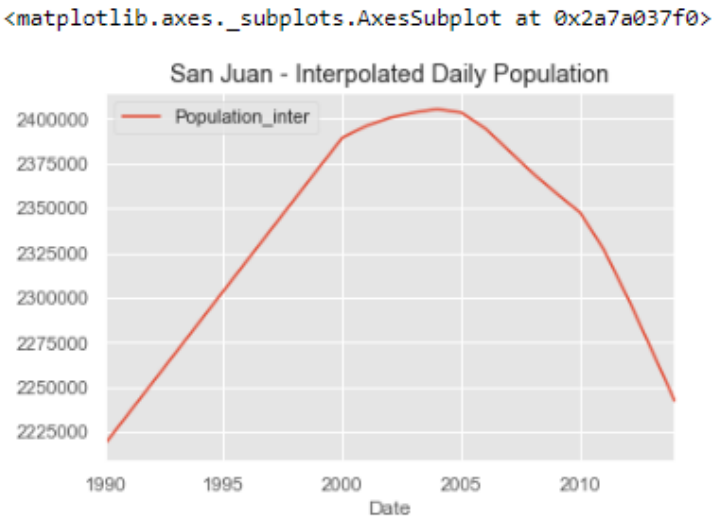


Figure 4.1C: Interpolated daily population of San Juan

4.1.2 Part 1C - Iquitos

Raw ‘'Iquitos_Population_Data.csv' was imported to plot Figure 4.1D. No missing data is seen between year 2000 – 2014 and the scatterplot can be fitted with a straight line. The same resampling method was used to increase the frequency from annual to daily in Figure 4.1E.

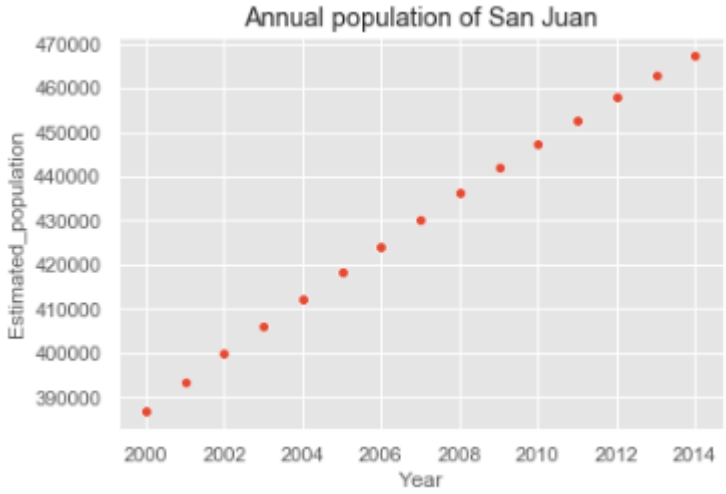


Figure 4.1D: Annual population of Iquitos

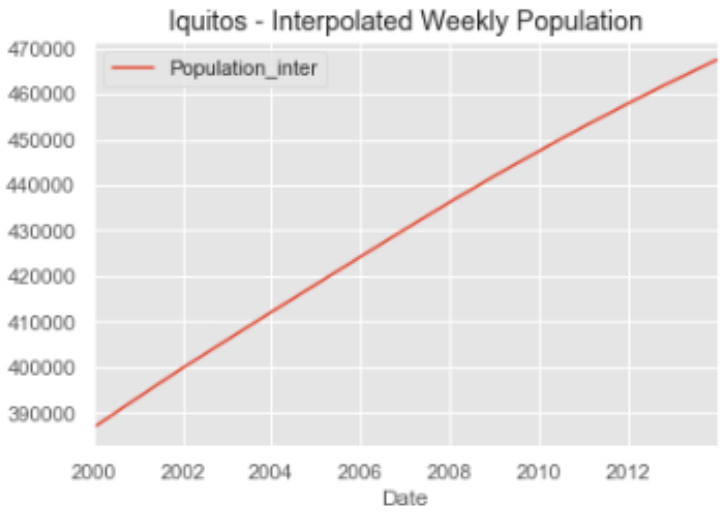


Figure 4.1E: Interpolated daily population of Iquitos

4.1.3 Part 1D - Merge San Juan & Iquitos interpolated daily populations datasets

The two populations were merged by rows and set to new unique identifier 'City_week_start_date' in Figure 4.1F.

```
<class 'pandas.core.frame.DataFrame'>
Index: 13882 entries, sj1989-12-31 to iq2013-12-31
Data columns (total 3 columns):
Date                13882 non-null datetime64[ns]
Population_inter    13882 non-null int32
city                13882 non-null object
dtypes: datetime64[ns](1), int32(1), object(1)
memory usage: 379.6+ KB
None

First 5 rows
                           Date  Population_inter city
City_week_start_date
sj1989-12-31         1989-12-31           2217968   sj
sj1990-01-01         1990-01-01           2218014   sj
sj1990-01-02         1990-01-02           2218061   sj
sj1990-01-03         1990-01-03           2218108   sj
sj1990-01-04         1990-01-04           2218155   sj

Last 5 rows
                           Date  Population_inter city
City_week_start_date
iq2013-12-27         2013-12-27            467441   iq
iq2013-12-28         2013-12-28            467454   iq
iq2013-12-29         2013-12-29            467467   iq
iq2013-12-30         2013-12-30            467480   iq
iq2013-12-31         2013-12-31            467493   iq
```

Figure 4.1F: 'pop_daily_combined' dataframe

4.2 Merge Population with Training & Test Datasets

Please refer to “PART 2 - Merge Population with Training & Test Datasets, Followed by Preprocessing” in attached Jupyter notebook.

4.2.1 Part 2B - Merge train_features and train_labels datasets

Two dataframe, i.e. ‘train_features’ and ‘train_labels’ were crated from ‘dengue_features_train.csv’ and ‘dengue_labels_train.csv’ respectively. The 'total_cases' column of ‘train_labels’ was added to ‘train_features’.

4.2.2 Part 2C - Merge train_features and test_features datasets

A ‘test_features’ dataframe was created from ‘dengue_features_test.csv’ before being concatenated by rows with ‘train_features’ to form ‘raw_dataset’.

4.2.3 Part 2D - Merge raw_dataset with pop_daily_combined

A ‘merged_raw_dataset’ dataframe was the inner join of ‘raw_dataset’ and ‘pop_daily_combined’ using the 'City_week_start_date' unique identifier. Create a new column, 'normal_total_cases' to normalize total_cases to be cases per 100,000 population using Population_inter of the same week.

4.2.4 Part 2E - Data Partition by City, Train vs. Test Datasets

Create new column, ‘Advance4W_normal_total_cases’ to carry forward 'normal_total_cases' 4 weeks in advance before splitting ‘merged_raw_dataset’ into sj and iq respectively. This rendered original train and test dataset segregation no longer be used since the last 4 rows of training dataset now has NaN ‘Advance4W_normal_total_cases’ from the test dataset.

Hence, each city dataframe were further split into train and test datasets depending on the availability of ‘Advance4W_normal_total_cases’. Rows with non-NaN ‘Advance4W_normal_total_cases’ are labelled as train while rows with NaN ‘Advance4W_normal_total_cases’ are labelled as test.

Table 4.1 lists the new train and test periods based on Figure 4.2A.

Table 4.1: Adjusted Train and Test Datasets

	San Juan	Iquitos
Train dataframe	sj_train	iq_train
Train row index	sj_index_train	iq_index_train
Train period	Start: 1990-04-30 End: 2008-03-25	Start: 2000-07-01 End: 2010-05-28
Test dataframe	sj_index_train	iq_test
Test row index	sj_index_test	iq_index_test
Test period	Start: 2008-04-01 End: 2013-04-23	Start: 2010-06-04 End: 2013-06-25

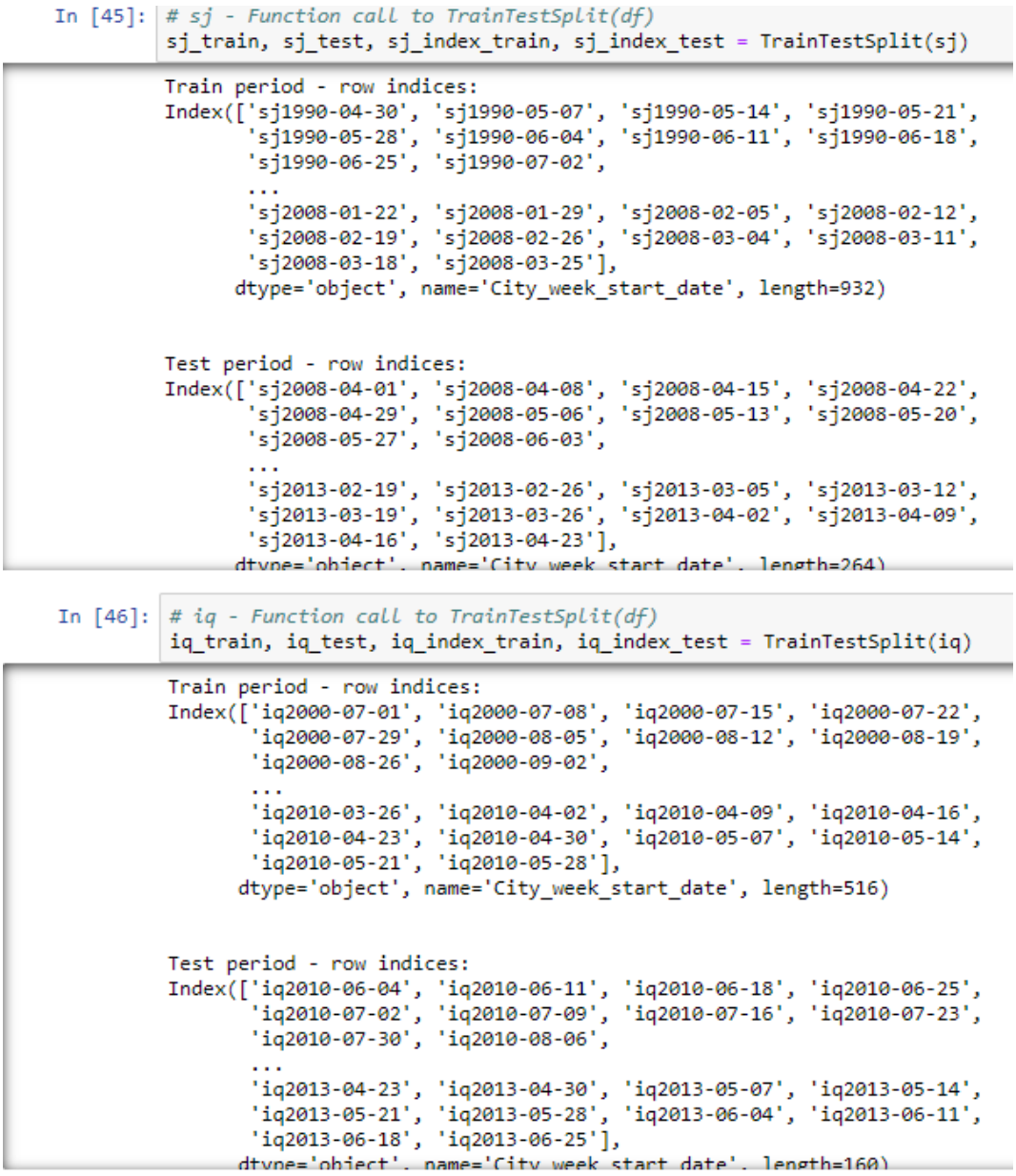


Figure 4.2A: Row indices of adjusted train and test datasets by city

4.2.5 Part 2F - Rescaling of variables (temperatures)

All temperature related variables measured using Kelvin were rescaled to Celcius using the formula in Figure 4.2B before being dropped from dataframes.

```
def rescaling(df) :
    """
    Objective:
        Convert all temperatures from Kelvin to Celsius
    Args:
        df: training dataset
    Returns:
        Nothing
    Raises:
        Nothing
    """

    # Create new columns for Celcius - minus 273.15 to convert from Kelvin
    df['normal_reanalysis_dew_point_temp_C'] = df['reanalysis_dew_point_temp_k'] - 273.15
    df['normal_reanalysis_max_air_temp_C']   = df['reanalysis_max_air_temp_k'] - 273.15
    df['normal_reanalysis_min_air_temp_C']   = df['reanalysis_min_air_temp_k'] - 273.15
    df['normal_reanalysis_avg_temp_C']       = df['reanalysis_avg_temp_k'] - 273.15
    df['normal_reanalysis_tdtr_C']           = df['reanalysis_tdtr_k'] - 273.15
    df['normal_reanalysis_air_temp_C']       = df['reanalysis_air_temp_k'] - 273.15

    # Delete original columns
    df.drop(['reanalysis_dew_point_temp_k', 'reanalysis_max_air_temp_k', 'reanalysis_min_air_temp_k',
             'reanalysis_avg_temp_k', 'reanalysis_tdtr_k', 'reanalysis_air_temp_k'],
            axis = 1, inplace = True)

    print(df.info())
    print("\nFirst 6 rows")
    print(df.head(6))
```

Figure 4.2B: Standardization of temperature variables

4.3 Explorative Data Analysis (EDA) of combined dataset

Please refer to “PART 3 - Explorative Data Analysis (EDA)” in attached Jupyter notebook.

4.3.1 Part 3B - Univariate Analysis

By comparing the mean of San Juan (Figure 4.3.1A) vs. Iquitos (Figure 4.3.1B), San Juan has

- ndvi - less vegetation
- precipitation - less rain fall
- humidity - less humid
- minumum temperature - higher
- maximum temperature - lower
- average temperature - higher
- diurnal temperature range (tdtr) - lower
- normal_total cases – less

San Juan is more urbanized with less rainfall, less humid, higher temperature and less critical dengue situation.

Descriptive statistics of sj_train

	year	weekofyear	ndvi_ne	ndvi_nw	ndvi_se	ndvi_sw	precipitation_amt_mm	reanalysis_precip_amt_kg_per_m2	reanalysis_relative_humidity_percent
count	932.00	932.00	741.00	884.00	913.00	913.00	923.00	926.00	926.00
mean	1998.79	26.55	0.06	0.07	0.18	0.17	35.60	30.55	78.58
std	5.19	15.04	0.11	0.09	0.06	0.06	44.66	35.67	3.39
min	1990.00	1.00	-0.41	-0.46	-0.02	-0.06	0.00	0.00	66.74
25%	1994.00	13.00	0.01	0.02	0.14	0.13	0.00	10.90	76.27
50%	1999.00	27.00	0.06	0.07	0.18	0.17	20.96	21.35	78.67
75%	2003.00	40.00	0.11	0.12	0.21	0.20	52.32	37.08	80.98
max	2008.00	53.00	0.49	0.44	0.39	0.38	390.60	570.50	87.58

8 rows × 24 columns

Figure 4.3.1A: Descriptive statistics of sj_train

Descriptive statistics of iq_train

	year	weekofyear	ndvi_ne	ndvi_nw	ndvi_se	ndvi_sw	precipitation_amt_mm	reanalysis_precip_amt_kg_per_m2	reanalysis_relative_humidity_percent
count	516.00	516.00	513.00	513.00	513.00	513.00	512.00	512.00	512.00
mean	2004.96	26.53	0.26	0.24	0.25	0.27	64.22	57.36	88.61
std	2.90	15.09	0.08	0.08	0.08	0.09	35.34	50.04	7.60
min	2000.00	1.00	0.06	0.04	0.03	0.06	0.00	0.00	57.79
25%	2002.00	13.00	0.20	0.18	0.19	0.20	39.09	23.95	84.24
50%	2005.00	27.00	0.26	0.23	0.25	0.26	60.47	46.22	90.89
75%	2007.00	40.00	0.32	0.29	0.30	0.33	85.76	71.07	94.56
max	2010.00	53.00	0.51	0.45	0.54	0.55	210.83	362.03	98.61

8 rows × 24 columns

Figure 4.3.1B: Descriptive statistics of iq_train

Based on time series plot (Figure 4.3.1C) and top 5 normal_total_cases (Figure 4.3.1D),

- sj_train has about twice as long history than iq_train
- sj_train has the worst dengue outbreak in October-November 1994.
- Iquitos has one particularly serious outbreak in December 2004.

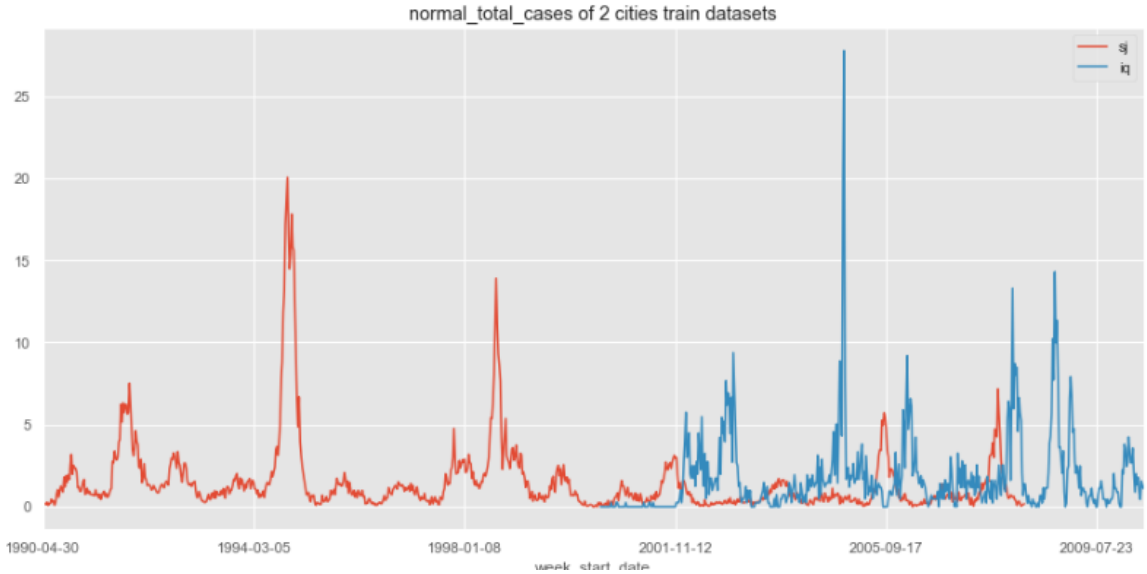


Figure 4.3.1C: normal_total_cases of 2 cities train datasets

```
# Check the rows for top 5 normal_total_cases
sj_train[['week_start_date', 'normal_total_cases']].nlargest(5, 'normal_total_cases', keep='first')
```

City_week_start_date	week_start_date	normal_total_cases
sj1994-10-15	1994-10-15	20.042886
sj1994-10-08	1994-10-08	18.523841
sj1994-11-12	1994-11-12	17.815382
sj1994-10-01	1994-10-01	17.178320
sj1994-10-22	1994-10-22	16.562366

```
# Check the rows for top 5 normal_total_cases
iq_train[['week_start_date', 'normal_total_cases']].nlargest(5, 'normal_total_cases', keep='first')
```

City_week_start_date	week_start_date	normal_total_cases
iq2004-12-09	2004-12-09	27.764348
iq2004-12-02	2004-12-02	19.871387
iq2008-10-14	2008-10-14	14.295050
iq2008-01-08	2008-01-08	13.293301
iq2008-10-28	2008-10-28	11.339617

Figure 4.3.1D: Top 5 highest weekly normal_total_cases of 2 cities train datasets

Based on Figure 4.3.1E, 'weekofyear' should be included as seasonality predictor as

- sj_train has lower 'normal_total_cases' during first half of the year.
- iq_train has lower 'normal_total_cases' during middle of the year.
- Dengue in Iquitos could be harder to predict due to the volatility toward last quarter of the year.

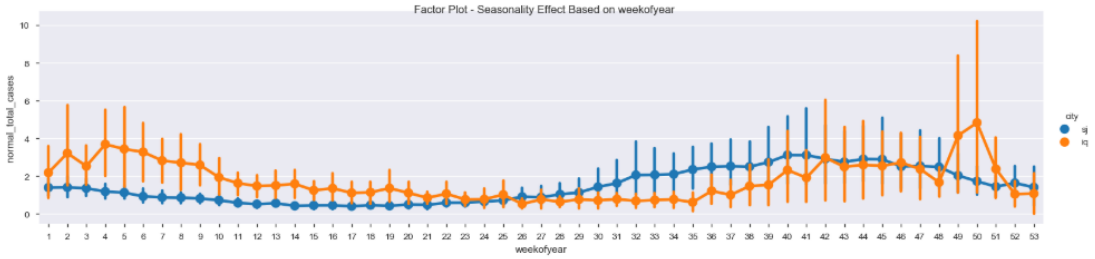


Figure 4.3.1E: Factor Plot by city for ‘weekofyear’ vs. ‘normal_total_cases’

The distributions of variables were examined via skewness statistics (Figure 4.3.1F) and histograms of Pandas Profiling (Figure 4.3.1 G-H). Based on San Juan's skewness and histogram (from pandas_profiling), these variables with skewness > +2 should undergo log transformation

- precipitation_amt_mm
- reanalysis_precip_amt_kg_per_m2
- reanalysis_sat_precip_amt_mm
- station_precip_mm
- normal_total_cases
- Advance4W_normal_total_cases

Skewness of San Juan

year	-0.000432
weekofyear	-0.006419
ndvi_ne	-0.021695
ndvi_nw	-0.092872
ndvi_se	0.214060
ndvi_sw	0.146674
precipitation_amt_mm	2.614021
reanalysis_precip_amt_kg_per_m2	5.566505
reanalysis_relative_humidity_percent	-0.198310
reanalysis_sat_precip_amt_mm	2.614021
reanalysis_specific_humidity_g_per_kg	-0.477818
station_avg_temp_c	-0.316232
station_diur_temp_rng_c	0.103033
station_max_temp_c	-0.445828
station_min_temp_c	-0.394211
station_precip_mm	2.627637
normal_reanalysis_dew_point_temp_C	-0.636283
normal_reanalysis_max_air_temp_C	-0.165481
normal_reanalysis_min_air_temp_C	-0.547910
normal_reanalysis_avg_temp_C	-0.232605
normal_reanalysis_tdtr_C	0.675330
normal_reanalysis_air_temp_C	-0.225818
normal_total_cases	4.509058
Advance4W_normal_total_cases	4.508416
dtype: float64	

Skewness of Iquitos

year	0.000182
weekofyear	-0.003026
ndvi_ne	0.228202
ndvi_nw	0.229115
ndvi_se	0.271795
ndvi_sw	0.273433
precipitation_amt_mm	0.600705
reanalysis_precip_amt_kg_per_m2	2.014553
reanalysis_relative_humidity_percent	-1.095952
reanalysis_sat_precip_amt_mm	0.600705
reanalysis_specific_humidity_g_per_kg	-0.661493
station_avg_temp_c	-0.906186
station_diur_temp_rng_c	-0.069135
station_max_temp_c	0.625411
station_min_temp_c	-1.146989
station_precip_mm	2.206239
normal_reanalysis_dew_point_temp_C	-0.882601
normal_reanalysis_max_air_temp_C	0.159096
normal_reanalysis_min_air_temp_C	-0.958635
normal_reanalysis_avg_temp_C	-0.112443
normal_reanalysis_tdtr_C	0.284781
normal_reanalysis_air_temp_C	0.102901
normal_total_cases	4.057032
Advance4W_normal_total_cases	4.064635
dtype: float64	

Figure 4.3.1F: Skewness of 2 cities train datasets

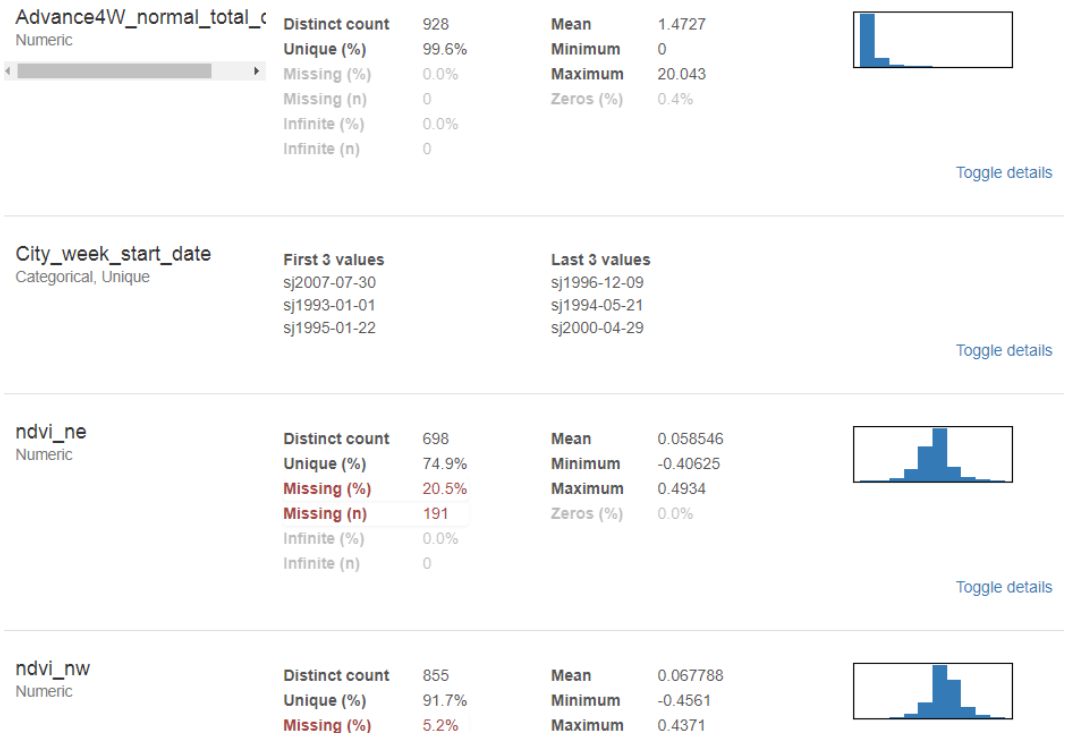


Figure 4.3.1G: Panda Profiling of sj_train (pfr_sj.html) - Histogram

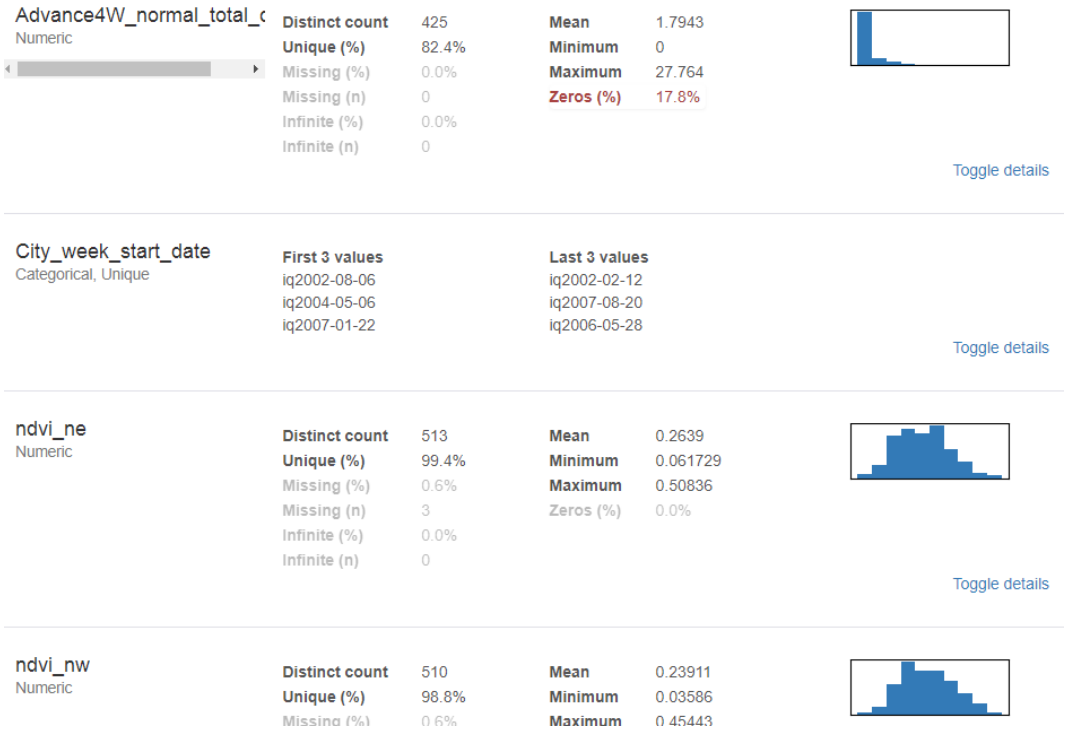
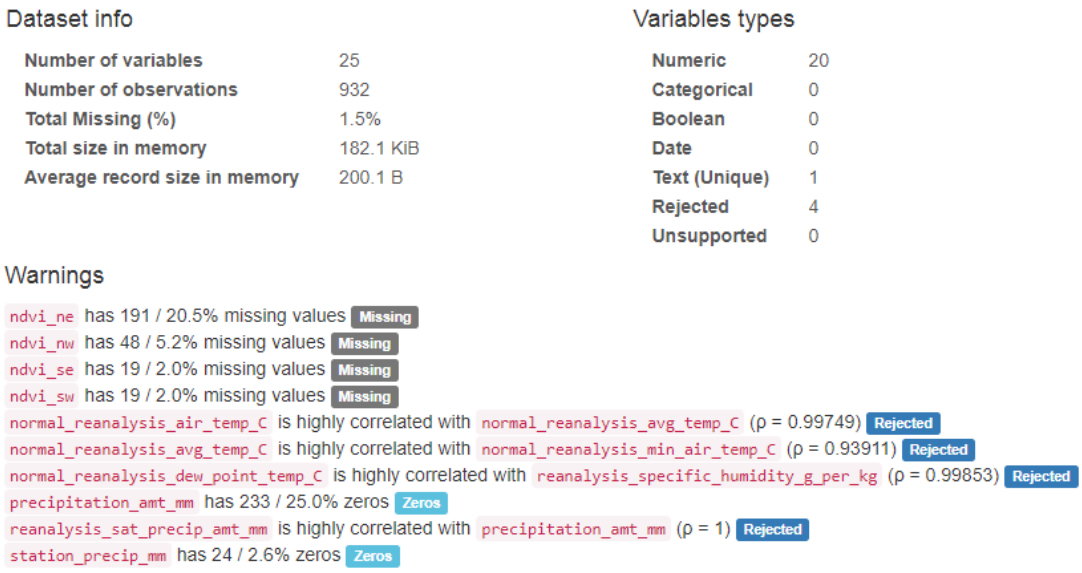


Figure 4.3.1H: Panda Profiling of iq_train (pfr_iq.html) - Histogram

4.3.2 Part 3C - Missing Values Analysis & Treatment

No variable has missing values more than 50% threshold in both cities based on Figure 4.3.2 A-B.



Dataset info

Number of variables	25
Number of observations	932
Total Missing (%)	1.5%
Total size in memory	182.1 KiB
Average record size in memory	200.1 B

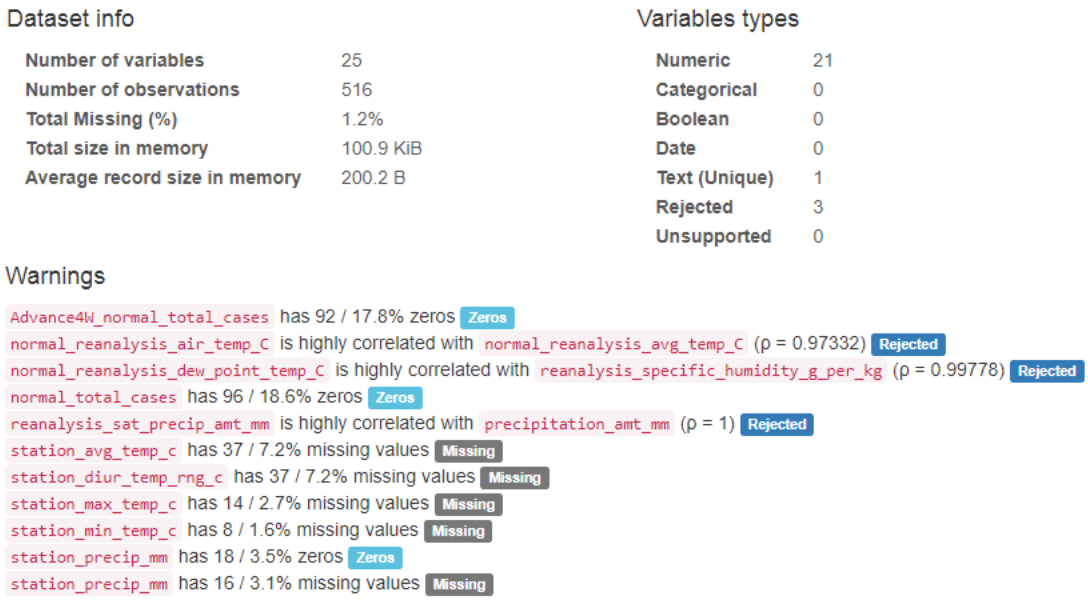
Variables types

Numeric	20
Categorical	0
Boolean	0
Date	0
Text (Unique)	1
Rejected	4
Unsupported	0

Warnings

- ndvi_ne has 191 / 20.5% missing values Missing
- ndvi_nw has 48 / 5.2% missing values Missing
- ndvi_se has 19 / 2.0% missing values Missing
- ndvi_sw has 19 / 2.0% missing values Missing
- normal_reanalysis_air_temp_C is highly correlated with normal_reanalysis_avg_temp_C (ρ = 0.99749) Rejected
- normal_reanalysis_avg_temp_C is highly correlated with normal_reanalysis_min_air_temp_C (ρ = 0.93911) Rejected
- normal_reanalysis_dew_point_temp_C is highly correlated with reanalysis_specific_humidity_g_per_kg (ρ = 0.99853) Rejected
- precipitation_amt_mm has 233 / 25.0% zeros Zeros
- reanalysis_sat_precip_amt_mm is highly correlated with precipitation_amt_mm (ρ = 1) Rejected
- station_precip_mm has 24 / 2.6% zeros Zeros

Figure 4.3.2A: Panda Profiling of sj_train (pfr_sj.html) – Missing values



Dataset info

Number of variables	25
Number of observations	516
Total Missing (%)	1.2%
Total size in memory	100.9 KiB
Average record size in memory	200.2 B

Variables types

Numeric	21
Categorical	0
Boolean	0
Date	0
Text (Unique)	1
Rejected	3
Unsupported	0

Warnings

- Advance4W_normal_total_cases has 92 / 17.8% zeros Zeros
- normal_reanalysis_air_temp_C is highly correlated with normal_reanalysis_avg_temp_C (ρ = 0.97332) Rejected
- normal_reanalysis_dew_point_temp_C is highly correlated with reanalysis_specific_humidity_g_per_kg (ρ = 0.99778) Rejected
- normal_total_cases has 96 / 18.6% zeros Zeros
- reanalysis_sat_precip_amt_mm is highly correlated with precipitation_amt_mm (ρ = 1) Rejected
- station_avg_temp_c has 37 / 7.2% missing values Missing
- station_diur_temp_rng_c has 37 / 7.2% missing values Missing
- station_max_temp_c has 14 / 2.7% missing values Missing
- station_min_temp_c has 8 / 1.6% missing values Missing
- station_precip_mm has 18 / 3.5% zeros Zeros
- station_precip_mm has 16 / 3.1% missing values Missing

Figure 4.3.2B: Panda Profiling of iq_train (pfr_iq.html) – Overview

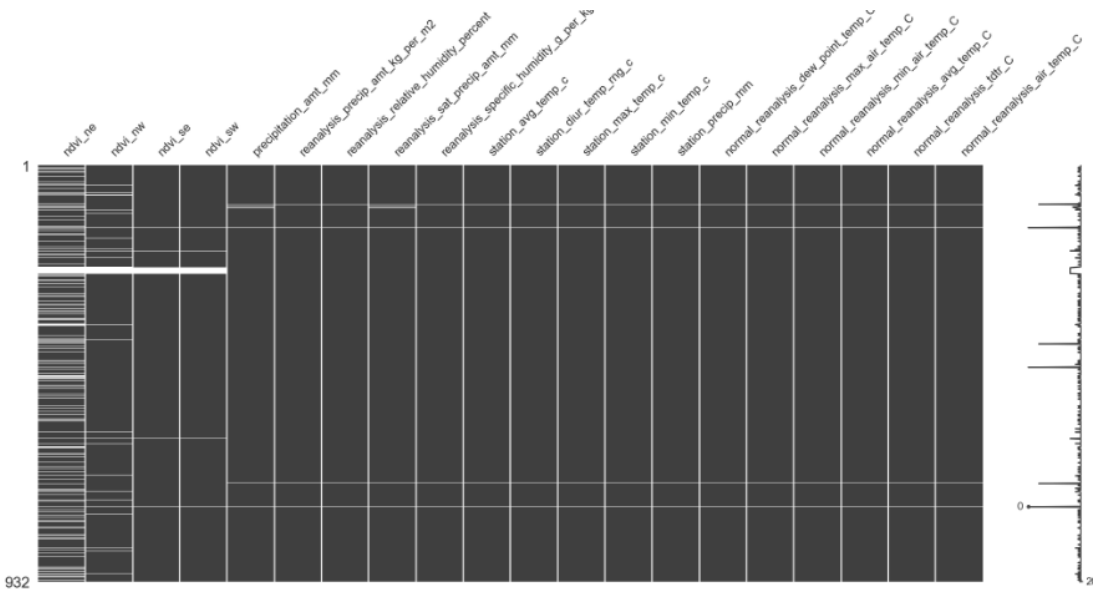


Figure 4.3.2C: Missing values matrix – sj_train

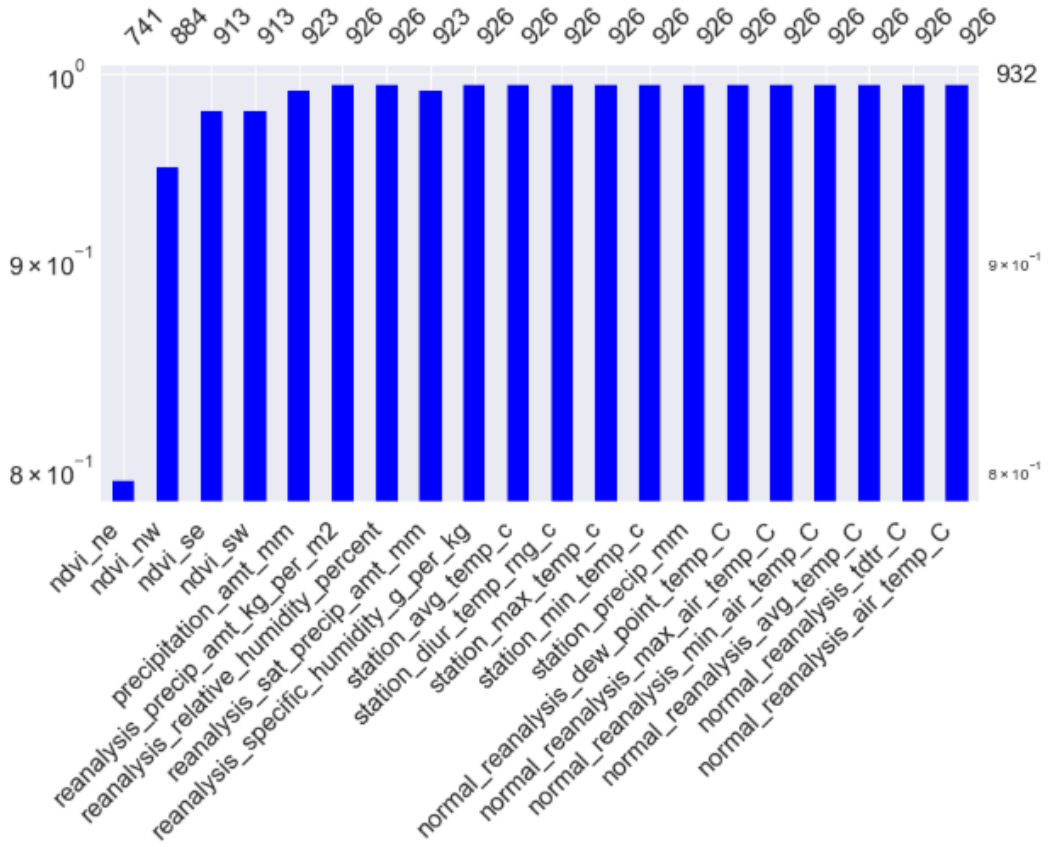


Figure 4.3.2D: Log bar chart of complete values – sj_train

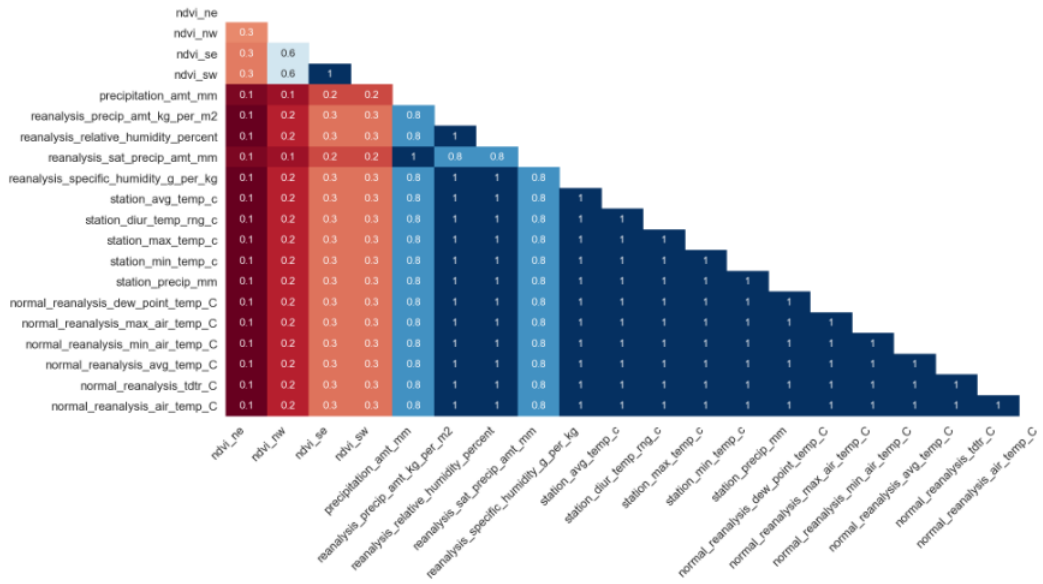


Figure 4.3.2E: Nullity Heatmap – sj_train

According to Akinfaderin (2017), nullity correlation (Figure 4.3.2E) ranges from -1 to +1. Nullity correlation of +1 means that the observations from the pair of features are missing concurrently while -1 means that the pair of observations does not have common missing observations.

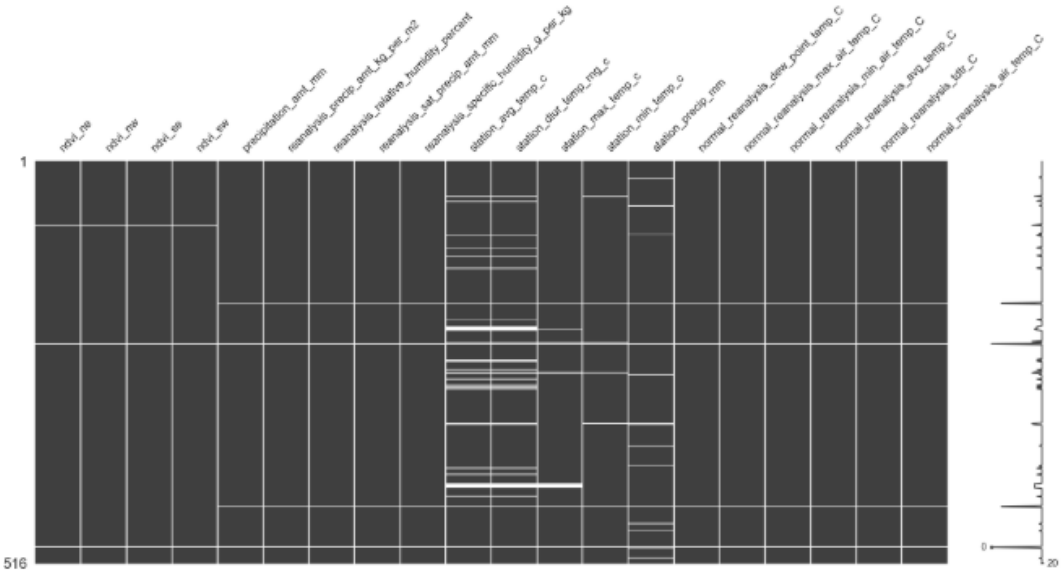


Figure 4.3.2F: Missing values matrix – iq_train

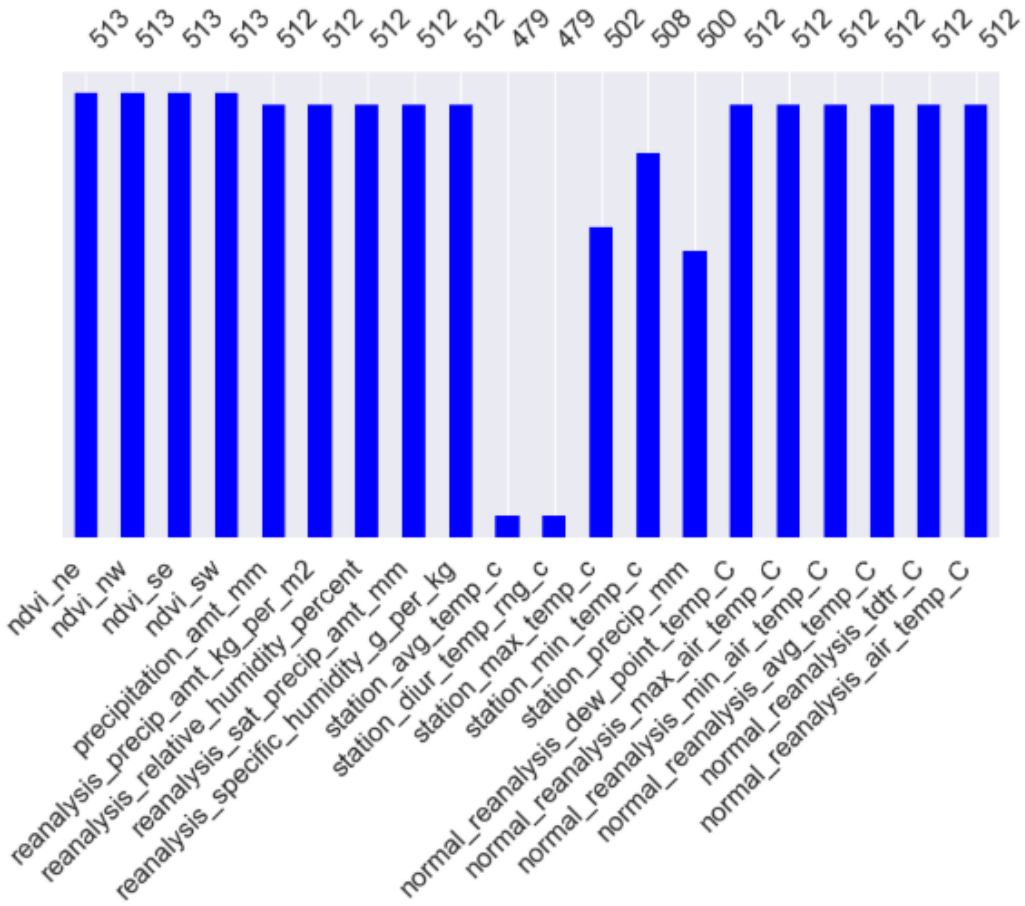


Figure 4.3.2G: Log bar chart of complete values – iq_train

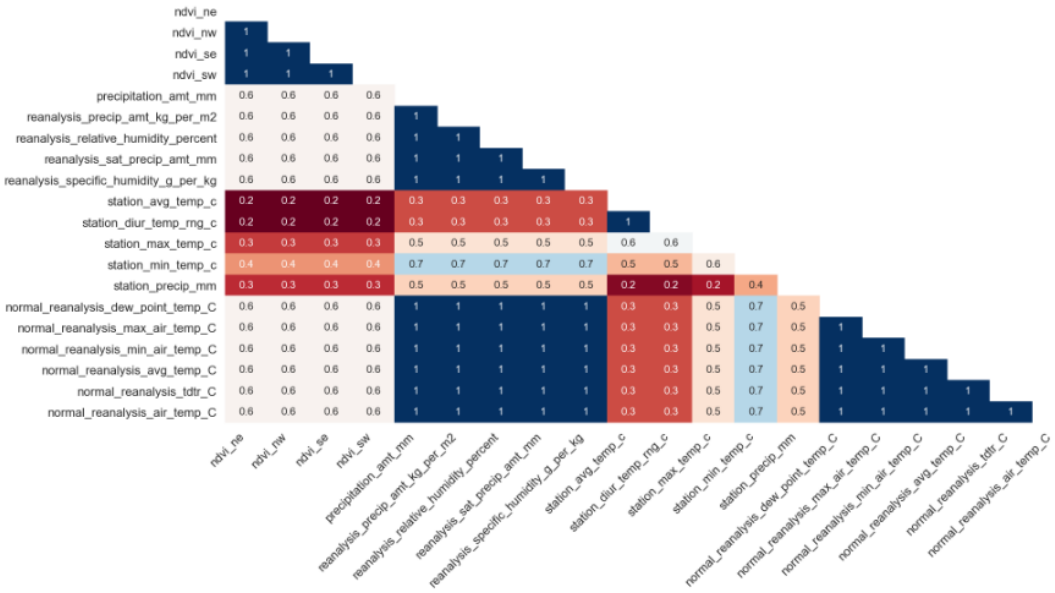


Figure 4.3.2H: Nullity Heatmap – iq_train

Based on Figures 4.3.2 A-H

1. Critical sources of missing values are

- San Juan's ndvi, esp. ndvi_ne
- Iquitos' weather station, esp. maximum & diural range temperatures

2. There are some periods whereby

- a. San Juan's ndvi_ne, ndvi_nw, ndvi_se, ndvi_sw were missing altogether.
- b. Iquitos' some weather station variables were missing.

3. There are some periods whereby the almost the variable set was missing in both cities.

4. Missing pattern from Nullity Heatmap

- Perfect nullity correlation between these variables is due to them having little missing values.

- a. San Juan - reanalysis & weather station variables
- b. Iquitos - reanalysis

Remedy for missing values

1. All variables have missing values but none is more than 50% missing. Hence, no variables will be excluded from the analysis.

2. There is no Missing Not At Random (MNAR). Thus, no dummy variable for missing value will be created.

3. Most of the missing values are Missing At Random (MCAR). Therefore, Multiple Imputation by Chained Equations (MICE) imputation (Figure 4.3.2I) is suitable in this case.

```
def MICE_imputation(df) :
    """ 1) Single imputation of MICE using fancyimpute.IterativeImputer 0.4.2
            a. https://stackoverflow.com/questions/45239256/data-imputation-with-fancyimpute-and-pandas
                - fancyimpute needs input as a numpay array.

            b. sample_posterior=True for multiple imputation with different random seed using iteration

        2) Convert filled series to dataframe before replacing missing value columns in df.
        3) Standardize the row index of imputed dataframe & non-numeric dataframe for overwrite.
        4) Reset to original 'City_week_start_date' index
        3) Null check to confirm there is no more missing values

    Args:
        df: numerical dataframe with missing values
    Returns:
        Imputed dataframe
    Raises:
        Nothing
    """

    # Convert df to series with no label before using IterativeImputer (which replaced the older MICE function)
    imputer    = IterativeImputer(n_iter = 50, sample_posterior = False, random_state = 123)
    df_missing = df[list_variables].values
    df_filled  = imputer.fit_transform(df_missing)

    # Convert back to dataframe with the correct column labels
    df_numeric = pd.DataFrame(data = df_filled, columns = list_variables)

    # Row index of df_numeric start from zero but sj_train & iq_train start is 'City_week_start_date'
    # Hence, need to reset sj_train & iq_train's row index to start from zero again in order to replace missing values for imput
    # But do not drop 'City_week_start_date' so that we can reset it back as index later
    df.reset_index(drop = False, inplace = True)

    # Overwrite the original columns with missing values by matching numeric row index
    df[list_variables] = df_numeric

    # Reset index back to unique identifier 'City_week_start_date'
    df.set_index('City_week_start_date', inplace = True)

    # print("Null check for number of missing values per column")
    # print(df.isnull().sum())
    print(df.info())
```

Figure 4.3.2I: MICE imputation

There are no more missing value after MICE imputation for sj_train (Figure 4.3.2J) and iq_train (Figure 4.3.2K).

```
<class 'pandas.core.frame.DataFrame'>
Index: 932 entries, sj1990-04-30 to sj2008-03-25
Data columns (total 29 columns):
year                                     932 non-null float64
weekofyear                               932 non-null float64
week_start_date                          932 non-null object
ndvi_ne                                  932 non-null float64
ndvi_nw                                  932 non-null float64
ndvi_se                                  932 non-null float64
ndvi_sw                                  932 non-null float64
precipitation_amt_mm                     932 non-null float64
reanalysis_precip_amt_kg_per_m2          932 non-null float64
reanalysis_relative_humidity_percent     932 non-null float64
reanalysis_sat_precip_amt_mm             932 non-null float64
reanalysis_specific_humidity_g_per_kg    932 non-null float64
station_avg_temp_c                       932 non-null float64
station_diur_temp_rng_c                  932 non-null float64
station_max_temp_c                       932 non-null float64
station_min_temp_c                       932 non-null float64
station_precip_mm                        932 non-null float64
total_cases                              932 non-null float64
dataset_label                            932 non-null object
Population_inter                         932 non-null int32
city                                     932 non-null object
normal_total_cases                       932 non-null float64
Advance4W_normal_total_cases             932 non-null float64
normal_reanalysis_dew_point_temp_C       932 non-null float64
normal_reanalysis_max_air_temp_C         932 non-null float64
normal_reanalysis_min_air_temp_C         932 non-null float64
normal_reanalysis_avg_temp_C             932 non-null float64
normal_reanalysis_tdtr_C                 932 non-null float64
normal_reanalysis_air_temp_C             932 non-null float64
dtypes: float64(25), int32(1), object(3)
memory usage: 214.8+ KB
None
```

Figure 4.3.2J: MICE imputation – sj_train

```
<class 'pandas.core.frame.DataFrame'>
Index: 516 entries, iq2000-07-01 to iq2010-05-28
Data columns (total 29 columns):
year                                    516 non-null float64
weekofyear                              516 non-null float64
week_start_date                         516 non-null object
ndvi_ne                                 516 non-null float64
ndvi_nw                                 516 non-null float64
ndvi_se                                 516 non-null float64
ndvi_sw                                 516 non-null float64
precipitation_amt_mm                    516 non-null float64
reanalysis_precip_amt_kg_per_m2         516 non-null float64
reanalysis_relative_humidity_percent    516 non-null float64
reanalysis_sat_precip_amt_mm            516 non-null float64
reanalysis_specific_humidity_g_per_kg   516 non-null float64
station_avg_temp_c                      516 non-null float64
station_diur_temp_rng_c                 516 non-null float64
station_max_temp_c                      516 non-null float64
station_min_temp_c                      516 non-null float64
station_precip_mm                       516 non-null float64
total_cases                             516 non-null float64
dataset_label                           516 non-null object
Population_inter                        516 non-null int32
city                                    516 non-null object
normal_total_cases                      516 non-null float64
Advance4W_normal_total_cases            516 non-null float64
normal_reanalysis_dew_point_temp_C      516 non-null float64
normal_reanalysis_max_air_temp_C        516 non-null float64
normal_reanalysis_min_air_temp_C        516 non-null float64
normal_reanalysis_avg_temp_C            516 non-null float64
normal_reanalysis_tdtr_C                516 non-null float64
normal_reanalysis_air_temp_C            516 non-null float64
dtypes: float64(25), int32(1), object(3)
memory usage: 118.9+ KB
None
```

Figure 4.3.2K: MICE imputation – iq_train

4.3.3 Part 3D - Comparing similar climatic variables from 4 measurement systems

The climatic variables from different measurement systems are compared by side by side.

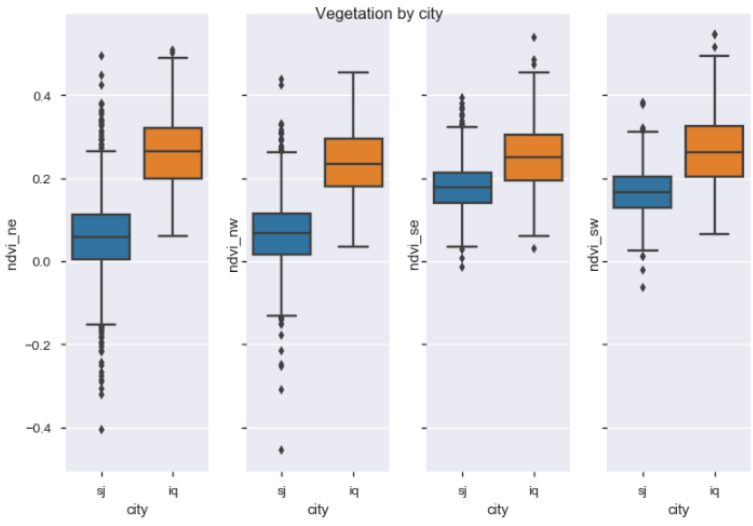


Figure 4.3.3A: Vegetation by city

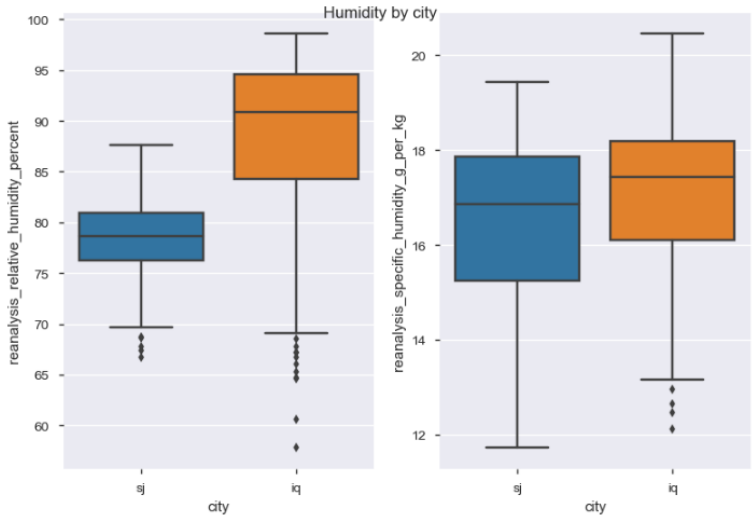


Figure 4.3.3B: Humidity by city

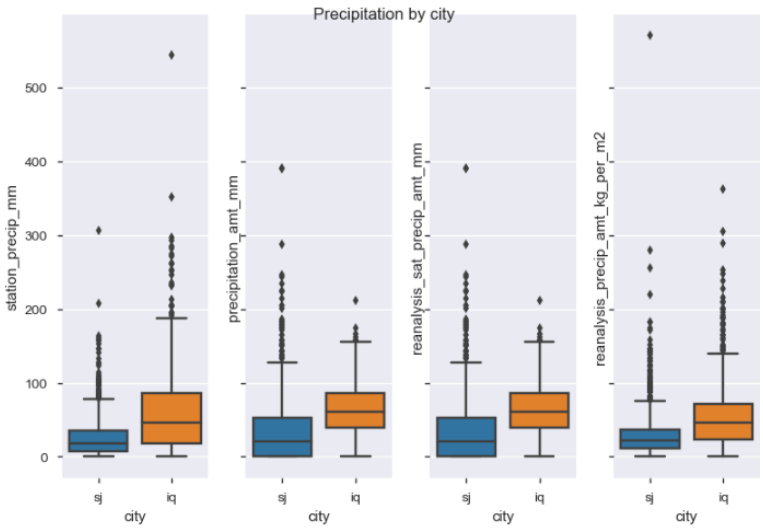


Figure 4.3.3C: Precipitation by city

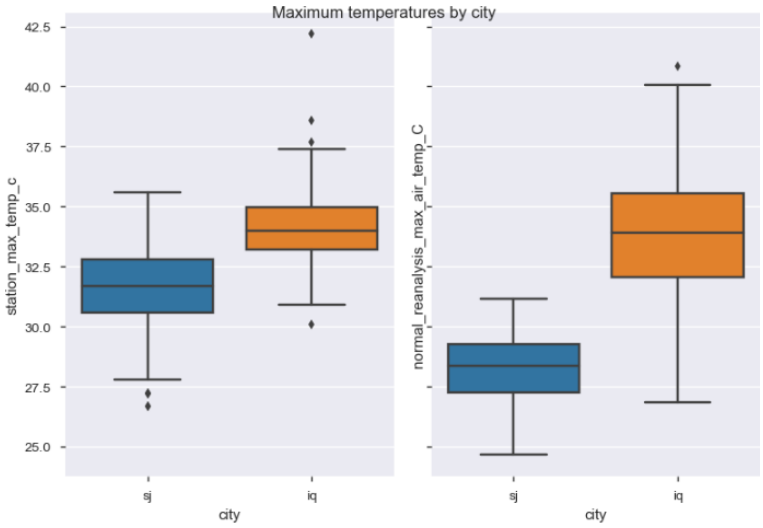


Figure 4.3.3D: Maximum temperatures by city

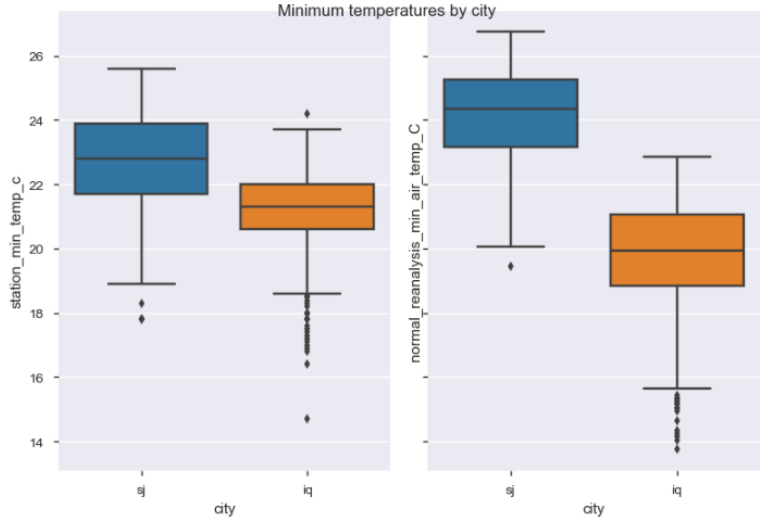


Figure 4.3.3E: Minimum temperatures by city

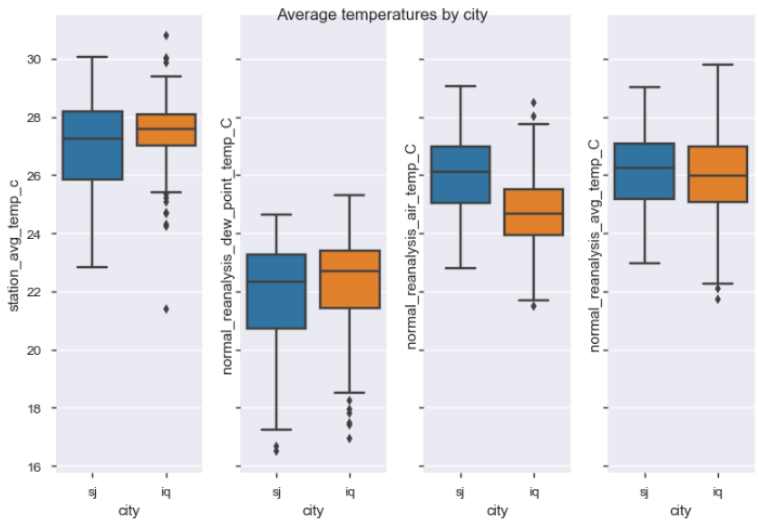


Figure 4.3.3F: Diural temperatures by city

There seem to be

1. Significant overlapped between the following measurements from different measurement systems could result in multi-collinearity

- weather station
- satellite precipitation
- reanalysis
- ndvi

2. Outliers exist for nearly climatic variables. They are assumed to be valid figures in the absence of wrong measurement evidence.

In conclusion, the different distributions of Advance4W_normal_total_cases and climatic variables for the 2 cities warrant separate forecasting models in nowcasting the dengue cases.

4.3.4 Part 3E - Correlation

<u>San Juan</u>

1. Based on Figure 4.3.2A: Panda Profiling, the following pairs of variables are highly correlated and the 1st mentioned should be rejected

- normal_reanalysis_air_temp_C vs. normal_reanalysis_avg_temp_C ($\rho = 0.9975$)

- normal_reanalysis_avg_temp_C vs. normal_reanalysis_min_air_temp_C (ρ = 0.93912)
- normal_reanalysis_dew_point_temp_C vs. reanalysis_specific_humidity_g_per_kg (ρ = 0.99853)
- reanalysis_sat_precip_amt_mm vs. precipitation_amt_mm (ρ = 1)

2. Based on Figure 4.3.4A: Pearson correlation heatmap, these variables have strong positive correlations

- ndvi_se vs. ndvi_sw
- reanalysis_specific_humidity_g_per_kg vs. station and reanalysis temperatures except tdtk (diural)
- station vs. reanalysis temperatures except tdtk (diural)
- normal_total_case vs. Advance4W_normal_total_case ¶

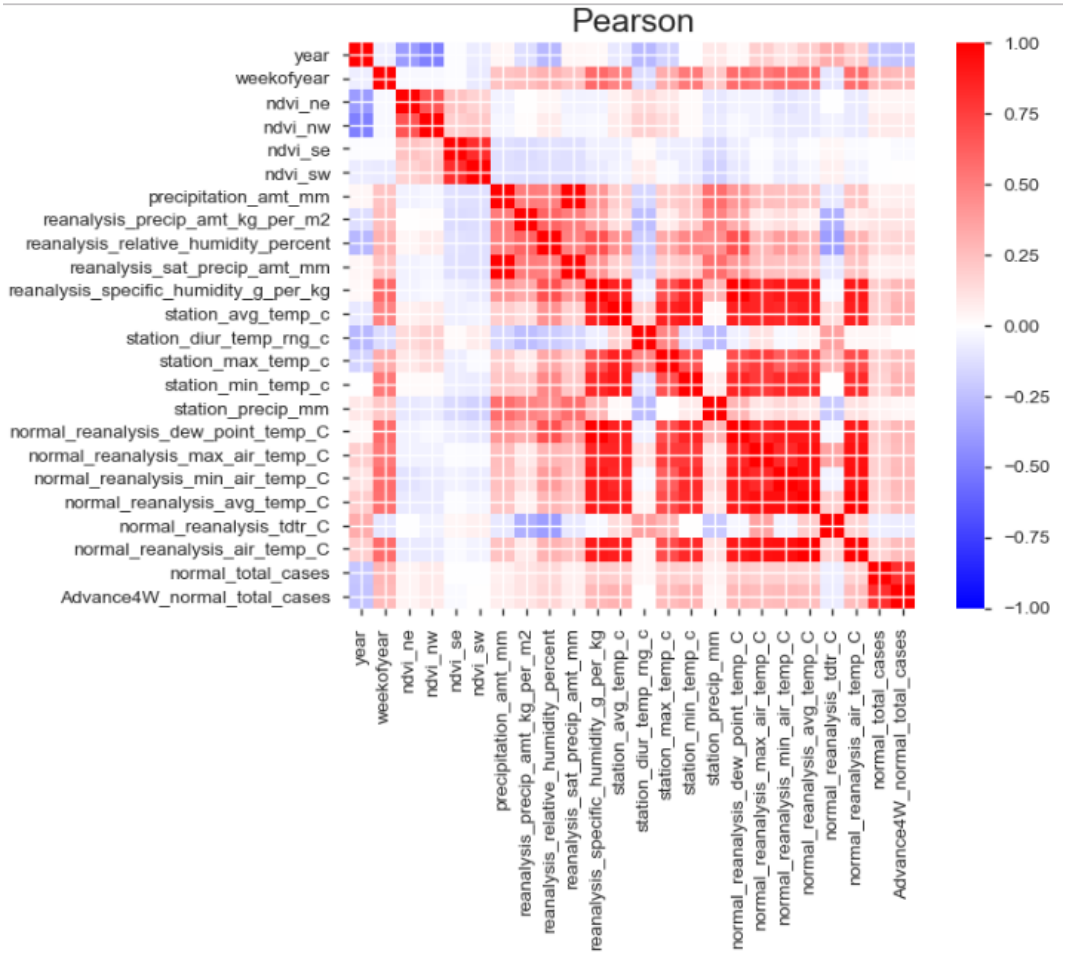


Figure 4.3.4A: Pearson correlation heatmap – sj_train

Iquitos

3. Based on Figure 4.3.2B: Panda Profiling, the following variables are highly correlated and the 1st mentioned should be rejected

- normal_reanalysis_air_temp_C vs. normal_reanalysis_avg_temp_C (ρ = 0.97332)
- normal_reanalysis_dew_point_temp_C vs. reanalysis_specific_humidity_g_per_kg (ρ = 0.99778)
- reanalysis_sat_precip_amt_mm vs. precipitation_amt_mm (ρ = 1)

4. Based on Figure 4.3.4B: Pearson correlation heatmap, these pairs of variables have strong positive correlations

- a. 4 sets of ndvi
- b. reanalysis_sat_precip_amt_mm vs. precipitation_amt_mm
- c. reanalysis_specific_humidity_g_per_kg vs. normal_reanalysis_dew_point_temp_C
- d. normal_reanalysis_air_temp_C vs. normal_reanalysis_avg_temp_C

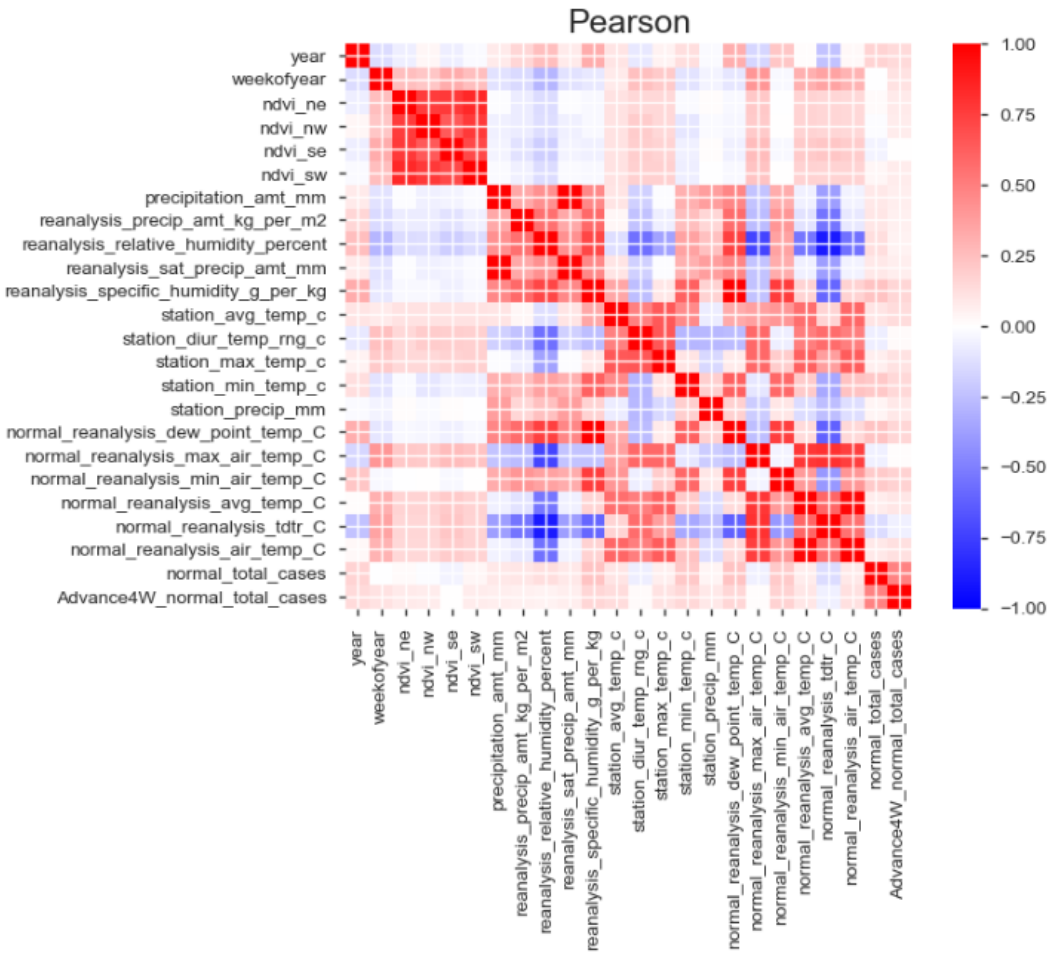


Figure 4.3.4B: Pearson correlation heatmap – iq_train

In conclusion, multicollinearity seems to be more serious among San Juan's variables than Iquitos'.

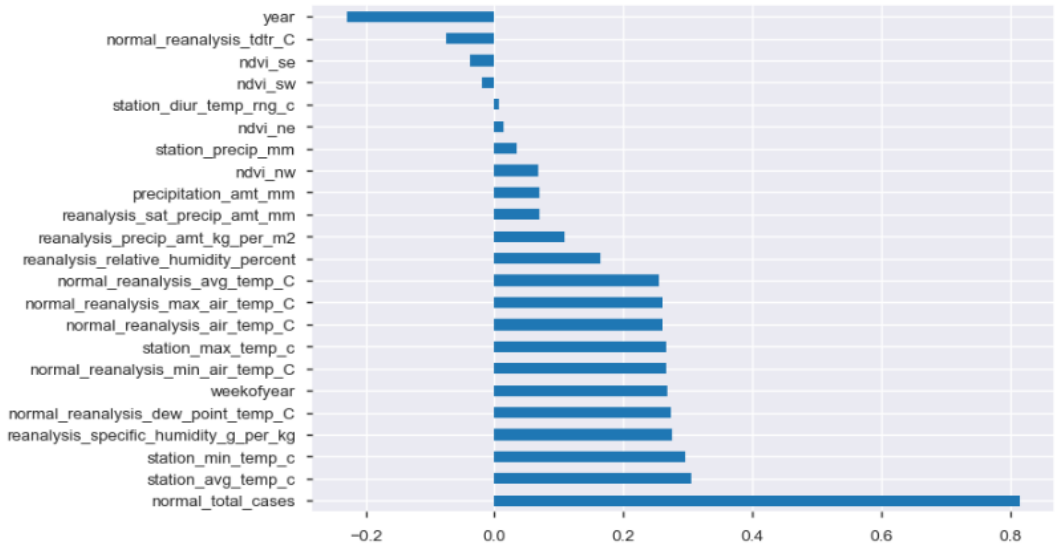


Figure 4.3.4C: Correlations between 'Advance4W_normal_total_cases' vs. predictors – sj_train

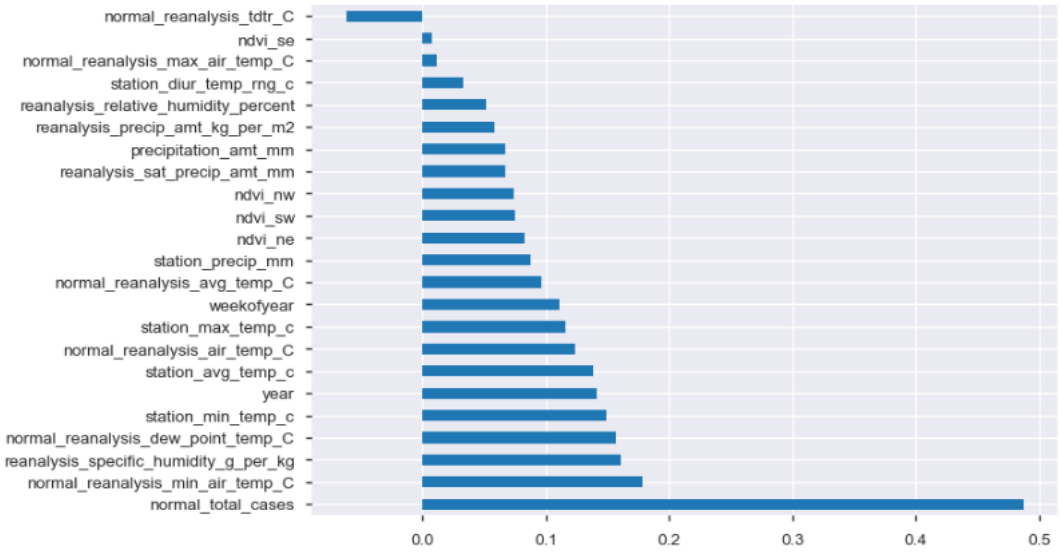


Figure 4.3.4D: Correlations between 'Advance4W_normal_total_cases' vs. predictors – iq_train

Based on Figure 4.3.4 C-D,

1. The climatic variables seem to have stronger impact in San Juan than Iquitos, based on the magnitude of the correlation coefficients.

2. Apart from normal_total_cases which have moderate to strong positive correlation with 'Advance4W_normal_total_cases', the correlations with other variables are quite weak.

3. Temperature related variables have the highest correlations compared to other climatic variables in both cities.

4. ndvi_se surprisingly has the lowest correlations compared to ndvi of other directions.

4.4 Feature Selection and Feature Engineering

Please refer to "PART 4 - Feature Selection and Feature Engineering" in attached Jupyter notebook.

4.4.1 PART 4A - Feature Selection (Round 1 - Based on EDA)

The following variables were selected based on EDA findings:

1. sj_train (8 variables)
   - year, ndvi_ne, ndvi_sw, reanalysis_sat_precip_amt_mm, normal_reanalysis_dew_point_temp_C, normal_reanalysis_avg_temp_C, normal_reanalysis_air_temp_C, normal_total_cases

2. iq_train (7 variables)
   - year, ndvi_nw, ndvi_se, reanalysis_sat_precip_amt_mm, normal_reanalysis_dew_point_temp_C, normal_reanalysis_air_temp_C, normal_total_cases

```
# sj_train - Drop variables which are unncessary or with strong correlations among predictors
sj_train.drop(['year', 'ndvi_ne', 'ndvi_sw', 'reanalysis_sat_precip_amt_mm', 'normal_reanalysis_dew_point_temp_C',
               'normal_reanalysis_avg_temp_C', 'normal_reanalysis_air_temp_C', 'normal_total_cases'], axis=1, inplace = True)

sj_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 932 entries, sj1990-04-30 to sj2008-03-25
Data columns (total 21 columns):
weekofyear                              932 non-null float64
week_start_date                         932 non-null object
ndvi_nw                                 932 non-null float64
ndvi_se                                 932 non-null float64
precipitation_amt_mm                    932 non-null float64
reanalysis_precip_amt_kg_per_m2         932 non-null float64
reanalysis_relative_humidity_percent    932 non-null float64
reanalysis_specific_humidity_g_per_kg   932 non-null float64
station_avg_temp_c                      932 non-null float64
station_diur_temp_rng_c                 932 non-null float64
station_max_temp_c                      932 non-null float64
station_min_temp_c                      932 non-null float64
station_precip_mm                       932 non-null float64
total_cases                             932 non-null float64
dataset_label                           932 non-null object
Population_inter                        932 non-null int32
city                                    932 non-null object
Advance4W_normal_total_cases            932 non-null float64
normal_reanalysis_max_air_temp_C        932 non-null float64
normal_reanalysis_min_air_temp_C        932 non-null float64
normal_reanalysis_tdtr_C                932 non-null float64
dtypes: float64(17), int32(1), object(3)
memory usage: 156.5+ KB
```

Figure 4.4.1A: Shortlisted variables – sj_train

```
# iq_train - Drop variables which are unncessary or with strong correlations among predictors
iq_train.drop(['year', 'ndvi_nw', 'ndvi_se', 'reanalysis_sat_precip_amt_mm', 'normal_reanalysis_dew_point_temp_C',
               'normal_reanalysis_air_temp_C', 'normal_total_cases'], axis=1, inplace = True)

iq_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 516 entries, iq2000-07-01 to iq2010-05-28
Data columns (total 22 columns):
weekofyear                              516 non-null float64
week_start_date                         516 non-null object
ndvi_ne                                 516 non-null float64
ndvi_sw                                 516 non-null float64
precipitation_amt_mm                    516 non-null float64
reanalysis_precip_amt_kg_per_m2         516 non-null float64
reanalysis_relative_humidity_percent    516 non-null float64
reanalysis_specific_humidity_g_per_kg   516 non-null float64
station_avg_temp_c                      516 non-null float64
station_diur_temp_rng_c                 516 non-null float64
station_max_temp_c                      516 non-null float64
station_min_temp_c                      516 non-null float64
station_precip_mm                       516 non-null float64
total_cases                             516 non-null float64
dataset_label                           516 non-null object
Population_inter                        516 non-null int32
city                                    516 non-null object
Advance4W_normal_total_cases            516 non-null float64
normal_reanalysis_max_air_temp_C        516 non-null float64
normal_reanalysis_min_air_temp_C        516 non-null float64
normal_reanalysis_avg_temp_C            516 non-null float64
normal_reanalysis_tdtr_C                516 non-null float64
dtypes: float64(18), int32(1), object(3)
memory usage: 90.7+ KB
```

Figure 4.4.1B: Shortlisted variables – iq_train

4.4.2 PART 4B - Logarithmic Transformation

The right-skewed variables with skewness more than +2 underwent ln(1+x) transformation to reduce the skewness.

```
# sj_train - Logarithmn transformation of right-skewed variables, including dependent variables
List_LogTrans_sj = ['precipitation_amt_mm', 'reanalysis_precip_amt_kg_per_m2', 'station_precip_mm',
                    'Advance4W_normal_total_cases']

def LogTransformation_sj(df, TestBoolean) :
    """
    Objectives:
        1) To transform right-skewed distributions in sj_train using ln(1+x)
        2) Ideally to fully automate the naming of the new variables using loop on List_LogTrans_sj
    Args:
        df:           contains right-skewed variables
        TestBoolean:  whether we're working with sj_train or sj_test
    Returns:
        None
    Raises:
        Nothing
    """

    df['LN_precipitation_amt_mm']           = df['precipitation_amt_mm'].apply(lambda x: np.log(x+1))
    df['LN_reanalysis_precip_amt_kg_per_m2'] = df['reanalysis_precip_amt_kg_per_m2'].apply(lambda x: np.log(x+1))
    df['LN_station_precip_mm']              = df['station_precip_mm'].apply(lambda x: np.log(x+1))

    # Exclude this step for sj_test
    if TestBoolean == False :
        df['LN_Advance4W_normal_total_cases']   = df['Advance4W_normal_total_cases'].apply(lambda x: np.log(x+1))
    elif TestBoolean == True :
        df['LN_Advance4W_normal_total_cases']   = np.NaN

    # Manual check confirmed that the calculation for the 1st 5 rows for 'LN_precipitation_amt_mm',
    #   'LN_reanalysis_precip_amt_kg_per_m2' and 'LN_Advance4W_normal_total_cases' were done correctly
    df.info()
    print("\n")
    print(df.head())
```

Figure 4.4.2A: Logarithmic transformation – sj_train

```
# sj_train - Logarithmn transformation of right-skewed variables, including dependent variables
List_LogTrans_sj = ['precipitation_amt_mm', 'reanalysis_precip_amt_kg_per_m2', 'station_precip_mm',
                    'Advance4W_normal_total_cases']

def LogTransformation_sj(df, TestBoolean) :
    """
    Objectives:
        1) To transform right-skewed distributions in sj_train using ln(1+x)
        2) Ideally to fully automate the naming of the new variables using loop on List_LogTrans_sj
    Args:
        df:           contains right-skewed variables
        TestBoolean:  whether we're working with sj_train or sj_test
    Returns:
        None
    Raises:
        Nothing
    """

    df['LN_precipitation_amt_mm']           = df['precipitation_amt_mm'].apply(lambda x: np.log(x+1))
    df['LN_reanalysis_precip_amt_kg_per_m2'] = df['reanalysis_precip_amt_kg_per_m2'].apply(lambda x: np.log(x+1))
    df['LN_station_precip_mm']              = df['station_precip_mm'].apply(lambda x: np.log(x+1))

    # Exclude this step for sj_test
    if TestBoolean == False :
        df['LN_Advance4W_normal_total_cases']   = df['Advance4W_normal_total_cases'].apply(lambda x: np.log(x+1))
    elif TestBoolean == True :
        df['LN_Advance4W_normal_total_cases']   = np.NaN

    # Manual check confirmed that the calculation for the 1st 5 rows for 'LN_precipitation_amt_mm',
    #   'LN_reanalysis_precip_amt_kg_per_m2' and 'LN_Advance4W_normal_total_cases' were done correctly
    df.info()
    print("\n")
    print(df.head())
```

Figure 4.4.2B: Logarithmic transformation – iq_train

Correlation coefficients were re-analysed between shortlisted transformed variables against 'LN_Advance4W_normal_total_cases'.

```
# Shortlisted variables - 15 variables
list_shortlist_sj = ['weekofyear', 'ndvi_nw', 'ndvi_se', 'LN_precipitation_amt_mm',
                     'LN_reanalysis_precip_amt_kg_per_m2', 'reanalysis_relative_humidity_percent',
                     'reanalysis_specific_humidity_g_per_kg', 'station_avg_temp_c', 'station_diur_temp_rng_c',
                     'station_max_temp_c', 'station_min_temp_c', 'LN_station_precip_mm',
                     'normal_reanalysis_max_air_temp_C', 'normal_reanalysis_min_air_temp_C', 'normal_reanalysis_tdtr_C']

# sj_train - Function call to Corr_LN_Advance4W_normal_total_cases (df, list_shortlist)
Corr_LN_Advance4W_normal_total_cases (sj_train, list_shortlist_sj)
```

```
Correlation coefficient between 'LN_Advance4W_normal_total_cases' vs. Predictors
```

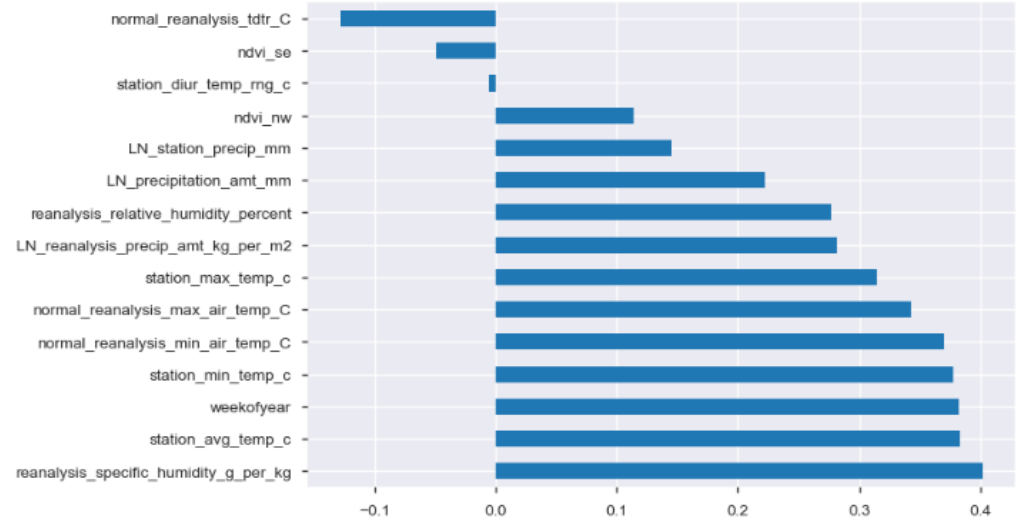


Figure 4.4.2C: Correlation coefficient between 'LN_Advance4W_normal_total_cases' vs. Predictors– sj_train

```
# Shortlisted variables - 16 variables
list_shortlist_iq = ['weekofyear', 'ndvi_ne', 'ndvi_sw', 'precipitation_amt_mm', 'LN_reanalysis_precip_amt_kg_per_m2',
                     'reanalysis_relative_humidity_percent', 'reanalysis_specific_humidity_g_per_kg', 'station_avg_temp_c',
                     'station_diur_temp_rng_c', 'station_max_temp_c', 'station_min_temp_c', 'LN_station_precip_mm',
                     'normal_reanalysis_max_air_temp_C', 'normal_reanalysis_min_air_temp_C',
                     'normal_reanalysis_avg_temp_C', 'normal_reanalysis_tdtr_C']

# sj_train - Function call to Corr_LN_Advance4W_normal_total_cases (df, list_shortlist)
Corr_LN_Advance4W_normal_total_cases (iq_train, list_shortlist_iq)
```

```
Correlation coefficient between 'LN_Advance4W_normal_total_cases' vs. Predictors
```

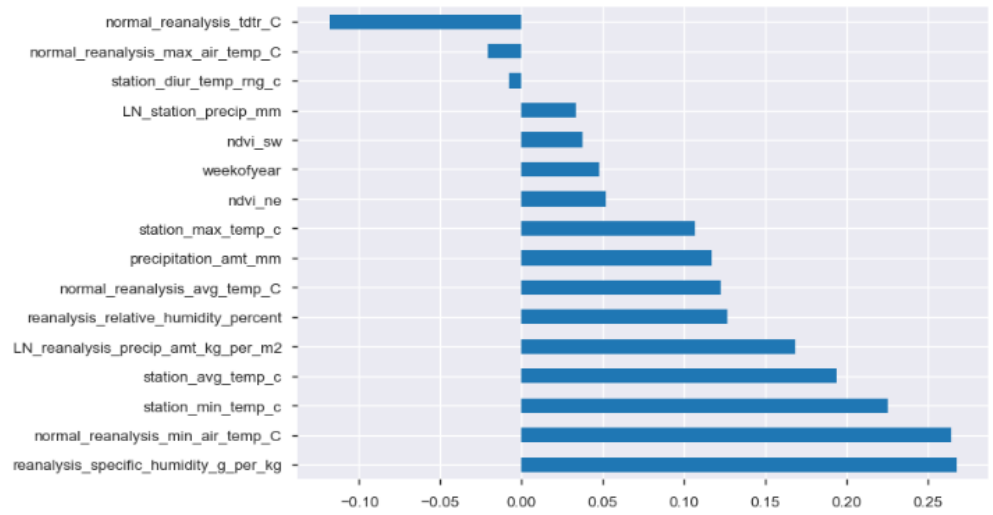


Figure 4.4.2D: Correlation coefficient between 'LN_Advance4W_normal_total_cases' vs. Predictors– iq_train

The Lasso model should be constructed using the transformed variables since the range of correlation coefficients between independent and dependent variables increased as per Table 4.2.

Table 4.2: Compare the effect of transformation on correlation coefficients
(excluding year & normal_total_cases)

City	sj_train	iq_train
Before transformation Coefficient range	Figure 4.3.4C -0.1 to +0.3	Figure 4.3.4D -0.1 to +0.2
Post transformation Coefficient range	Figure 4.4.2C -0.15 to + 0.4	Figure 4.4.2D -0.12 to + 0.26

4.4.3 PART 4C - Standardization Transformation

Features underwent standardization (minus mean and divided by standard deviation) before filtering using VarianceThreshold. The means are zeros while the standard deviations are ones in Figure 4.4.3 A-B.

```
# sj_train - Function call for Standardization(df, list_df)
Standardization(sj_train, list_shortlist_sj)

# Descriptive statistics, round to 2 decimal places
# The output should have zero means & 1 standard deviations
round(sj_train[list_shortlist_sj].describe(), 2)
```

	weekofyear	ndvi_nw	ndvi_se	LN_precipitation_amt_mm	LN_reanalysis_precip_amt_kg_per_m2	reanalysis_relative_humidity_percent	reanalysis_specific_
count	932.00	932.00	932.00	932.00	932.00	932.00	
mean	0.00	0.00	-0.00	-0.00	0.00	-0.00	
std	1.00	1.00	1.00	1.00	1.00	1.00	
min	-1.70	-5.78	-3.41	-1.51	-3.14	-3.51	
25%	-0.90	-0.55	-0.66	-1.34	-0.57	-0.67	
50%	0.03	0.01	-0.02	0.30	0.08	0.02	
75%	0.89	0.52	0.60	0.79	0.64	0.70	
max	1.76	4.07	3.81	1.95	3.44	2.67	

Figure 4.4.3A: Standardization of independent variables – sj_train

```
# iq_train - Function call for Standardization(df, list_df)
Standardization(iq_train, list_shortlist_iq)

# Descriptive statistics, round to 2 decimal places
# The output should have zero means & 1 standard deviations
round(iq_train[list_shortlist_iq].describe(), 2)
```

	weekofyear	ndvi_ne	ndvi_sw	precipitation_amt_mm	LN_reanalysis_precip_amt_kg_per_m2	reanalysis_relative_humidity_percent	reanalysis_specific_hum
count	516.00	516.00	516.00	516.00	516.00	516.00	
mean	-0.00	0.00	-0.00	-0.00	-0.00	-0.00	
std	1.00	1.00	1.00	1.00	1.00	1.00	
min	-1.69	-2.49	-2.35	-1.83	-3.78	-4.07	
25%	-0.90	-0.79	-0.73	-0.71	-0.48	-0.57	
50%	0.03	0.00	-0.05	-0.11	0.17	0.30	
75%	0.89	0.69	0.67	0.61	0.59	0.79	
max	1.76	3.01	3.24	4.17	2.26	1.32	

Figure 4.4.3B: Standardization of independent variables – iq_train

4.4.4 PART 4D - Feature Selection (Round 2 - Low Variance)

No feature was removed based on 0.7 variance threshold in Figure 4.4.4.

```
def variance_threshold_selector(df, list_df, threshold = 0.7):
    """
    Objectives:
        Check which column will be removed using variance threshold of 0.7
    Args:
        df:        contains standardized variables
        list:      list of all standardized variables to be evaluated
        threshold: default at 0.7
    Returns:
        Nothing
    Raises:
        Nothing
    """
    df5 = df[list_df]
    selector = VarianceThreshold(threshold)

    # Learn empirical variances from input
    selector.fit(df5)

    # Get the list of features selected
    list_remaining_column = df5.columns[selector.get_support(indices=True)]

    # Get the list of column removed by comparing original column list against the remaining
    #    Using set() from https://www.geeksforgeeks.org/python-difference-two-lists/
    list_drop_columns = list(set(list_df) - set(list_remaining_column))

    # If no variable is drop, then the list is empty
    if list_drop_columns == [] :
        print("No variable needs to be dropped.")
    else :
        print(list_drop_columns)
```

```
# sj_train - Function call for variance_threshold_selector(df, list_df, threshold = 0.7)
# Low variance threshold examination using standardized variables
variance_threshold_selector(sj_train, list_shortlist_sj)
```

```
No variable needs to be dropped.
```

```
# iq_train - Function call for variance_threshold_selector(df, list_df, threshold = 0.7)
variance_threshold_selector(iq_train, list_shortlist_iq)
```

```
No variable needs to be dropped.
```

Figure 4.4.4: Feature selection using VarianceThreshold

4.4.5 PART 4E - Cross Correlation Function (CCF) in R to Determine Lag Effects

The transformed datasets, sj_train and iq_train were exported to R via csv files.

```
# Export sj_train & iq_train into CSV format in order to be analyzed using CCF function in R
# Files will be saved in current working directory
sj_train.to_csv('sj_train.csv', encoding = 'utf-8')
iq_train.to_csv('iq_train.csv', encoding = 'utf-8')
```

Figure 4.4.5: Export dataframes into csv files

4.4.6 CCF Analysis in R

Please refer to the following R codes

- sj_train - Cross Correlation Function to Determine Significant Lags v2.R
- iq_train - Cross Correlation Function to Determine Significant Lags v2.R

```
# ------------------------------------------------------------------------------
# Part 2 - File importation & TS object
# ------------------------------------------------------------------------------

# Load csv file of train datasets files generated from C:\Users\ASUS PC\CP2 - v2.ipynb
sj_train <- read.csv(
  file="sj_train.csv", header=TRUE, stringsAsFactors = FALSE
)
```

Figure 4.4.6A: Create a dataframe from sj_train.csv

```
# Create time series object with weekly data
#  https://otexts.com/fpp2/ts-objects.html on creating weekly object
LN_Advance4W_normal_total_cases <- ts(sj_train$LN_Advance4W_normal_total_cases,
                                      frequency= 365.25 / 7, start=c(1990, 4, 30))
```

Figure 4.4.6B: Create time series object for dependent variable,

LN_Advance4W_normal_total_cases

```
# List of climatic variables (14)
# - Refer to Python's list_standardize_sj
list_climatic <- c('ndvi_nw', 'ndvi_se', 'LN_precipitation_amt_mm',
                   'LN_reanalysis_precip_amt_kg_per_m2',
                   'reanalysis_relative_humidity_percent',
                   'reanalysis_specific_humidity_g_per_kg', 'station_avg_temp_c',
                   'station_diur_temp_rng_c', 'station_max_temp_c',
                   'station_min_temp_c', 'LN_station_precip_mm',
                   'normal_reanalysis_max_air_temp_C',
                   'normal_reanalysis_min_air_temp_C', 'normal_reanalysis_tdtr_C')
```

Figure 4.4.6C: Create a list for climatic variables

```
# ------------------------------------------------------------------------------
# Part 3 - CCF for ndvi_nw
# ------------------------------------------------------------------------------

climatic <- list_climatic[[1]]

# Create ts object
ndvi_nw <- ts(sj_train[[climatic]], frequency= 365.25 / 7, start=c(1990, 4, 30))

# lag (the h in xt+h) and correlation with yt
ccfvalues_ndvi_nw <- ccf(ndvi_nw, LN_Advance4W_normal_total_cases, lag = 32)
ccfvalues_ndvi_nw

# Create CCF plot with 95% confidence interval with maximum lag of 32 weeks as per
#  - An ensemble prediction approach to weekly Dengue cases forecasting based
#    on climatic and terrain conditions
ccf(ndvi_nw, LN_Advance4W_normal_total_cases, lag = 32, correlation = TRUE, pl = TRUE,
    xlab = 'lags (fraction of a year)', ylab = 'CCF',

    main = paste("sj_train Cross Correlation - \n", climatic,
                 " vs LN_Advance4W_normal_total_cases"))

dev.copy(jpeg,filename=paste("sj_train CCF -", climatic, ".jpeg"))
dev.off()
```

Figure 4.4.6D: Generate CCF for ndvi_nw

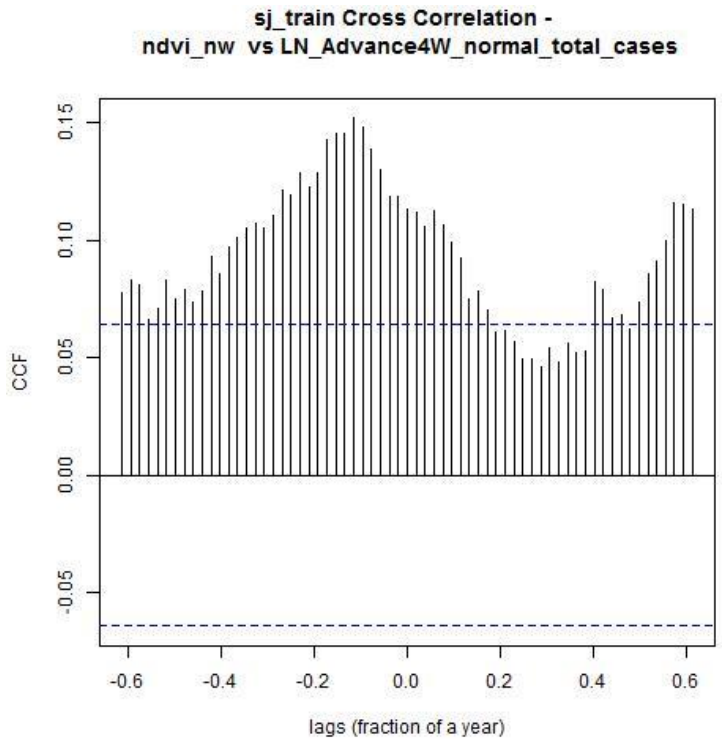


Figure 4.4.6E: CCF and 95% confidence interval for ndvi_nw

The weeks with lags were about outside of the 95% interval, i.e. significant CCF were noted in "Appendix 2 – Feature Selection Process for sj_train". Repeat the process above for all the climatic variables and paste the CCF charts in "Appendix 4 - Cross Correlation Functions from R".

4.5 PART 4F - Feature Engineering Based on CCF Analysis in R

Figure 4.5 A-B listed the climatic variables and their corresponding lags with significant CCF as per Appendix 4. The UDF in Figure 4.5C was used to create lagged climatic variables.

```
# sj_train - Dictionary containing climatic_var, start_lag, end_lag

                # ndvi_nw - weeks 0-32
dict_sj_train = {1: ['ndvi_nw', 0, 32],

                # ndvi_se - weeks 3, 5-21
                2: ['ndvi_se', 3, 3],
                3: ['ndvi_se', 5, 21],

                # LN_precipitation_amt_mm - weeks 19-32
                4: ['LN_precipitation_amt_mm', 19, 32],

                # LN_reanalysis_precip_amt_kg_per_m2 - weeks 0-13, 18-32
                5: ['LN_reanalysis_precip_amt_kg_per_m2', 0, 13],
                6: ['LN_reanalysis_precip_amt_kg_per_m2', 18, 32],

                # reanalysis_relative_humidity_percent - weeks 0-17, 23-32
                7: ['reanalysis_relative_humidity_percent', 0, 17],
                8: ['reanalysis_relative_humidity_percent', 23, 32],

                # reanalysis_specific_humidity_g_per_kg - weeks 0-12, 19-32
                9: ['reanalysis_specific_humidity_g_per_kg', 0, 12],
                10: ['reanalysis_specific_humidity_g_per_kg', 19, 32],

                # station_avg_temp_c - weeks 0-19, 22-32
                11: ['station_avg_temp_c', 0, 19],
                12: ['station_avg_temp_c', 22, 32],

                # station_diur_temp_rng_c - weeks 10-32
                13: ['station_diur_temp_rng_c', 10, 32],

                # station_max_temp_c - weeks 0-21, 25-32
                14: ['station_max_temp_c', 0, 21],
                15: ['station_max_temp_c', 25, 32],

                # station_min_temp_c - weeks 0-18, 21-32
                16: ['station_min_temp_c', 0, 18],
                17: ['station_min_temp_c', 21, 32],

                # LN_station_precip_mm - weeks 0-7, 14-32
                18: ['LN_station_precip_mm', 0, 7],
                19: ['LN_station_precip_mm', 14, 32],

                # normal_reanalysis_max_air_temp_C - weeks 0-16, 19-32
                20: ['normal_reanalysis_max_air_temp_C', 0, 16],
                21: ['normal_reanalysis_max_air_temp_C', 19, 32],

                # normal_reanalysis_min_air_temp_C - weeks 0-15, 19-32
                22: ['normal_reanalysis_min_air_temp_C', 0, 15],
                23: ['normal_reanalysis_min_air_temp_C', 19, 32],

                # normal_reanalysis_tdtr_C - weeks 0-8, 30-32
                24: ['normal_reanalysis_tdtr_C', 0, 8],
                25: ['normal_reanalysis_tdtr_C', 30, 32]}
```

Figure 4.5A: Dictionary containing climatic_var, start_lag, end_lag - sj_train

```
                # ndvi_ne - weeks 7-8, 10-11, 31-32
dict_iq_train = {1: ['ndvi_ne', 7, 8],
                 2: ['ndvi_ne', 10, 11],
                 3: ['ndvi_ne', 31, 32],

                 # ndvi_sw - weeks 4-18, 19-21, 31-32
                 4: ['ndvi_sw', 4, 18],
                 5: ['ndvi_sw', 19, 21],
                 6: ['ndvi_sw', 31, 32],

                 # precipitation_amt_mm - weeks 0-2, 9-13, 15-23
                 7: ['precipitation_amt_mm', 0, 2],
                 8: ['precipitation_amt_mm', 9, 13],
                 9: ['precipitation_amt_mm', 15, 23],

                 # LN_reanalysis_precip_amt_kg_per_m2 - weeks 0-5, 13, 17-19, 31
                 10: ['LN_reanalysis_precip_amt_kg_per_m2', 0, 5],
                 11: ['LN_reanalysis_precip_amt_kg_per_m2', 13, 13],
                 12: ['LN_reanalysis_precip_amt_kg_per_m2', 17, 19],
                 13: ['LN_reanalysis_precip_amt_kg_per_m2', 31, 31],

                 # reanalysis_relative_humidity_percent - weeks 0-2, 13, 27-32
                 14: ['reanalysis_relative_humidity_percent', 0, 2],
                 15: ['reanalysis_relative_humidity_percent', 13, 13],
                 16: ['reanalysis_relative_humidity_percent', 27, 32],

                 # reanalysis_specific_humidity_g_per_kg - weeks 0-8, 17-23
                 17: ['reanalysis_specific_humidity_g_per_kg', 0, 8],
                 18: ['reanalysis_specific_humidity_g_per_kg', 17, 23],
                 # station_avg_temp_c - weeks 0-9, 15, 18 - 32
                 19: ['station_avg_temp_c', 0, 9],
                 20: ['station_avg_temp_c', 15, 15],
                 21: ['station_avg_temp_c', 18, 32],

                 # station_diur_temp_rng_c - weeks 24-32
                 22: ['station_diur_temp_rng_c', 24, 32],

                 # station_max_temp_c - weeks 0-7, 22-32
                 23: ['station_max_temp_c', 0, 7],
                 24: ['station_max_temp_c', 22, 32],

                 # station_min_temp_c - weeks 0-5, 12-27
                 25: ['station_min_temp_c', 0, 5],
                 26: ['station_min_temp_c', 12, 27],

                 # LN_station_precip_mm - weeks 10-13, 18
                 27: ['LN_station_precip_mm', 10, 13],
                 28: ['LN_station_precip_mm', 18, 18],

                 # normal_reanalysis_max_air_temp_C - weeks 6-14, 25-32
                 29: ['normal_reanalysis_max_air_temp_C', 6, 14],
                 30: ['normal_reanalysis_max_air_temp_C', 25, 32],

                 # normal_reanalysis_min_air_temp_C - weeks 0-7, 13, 15, 17-28
                 31: ['normal_reanalysis_min_air_temp_C', 0, 7],
                 32: ['normal_reanalysis_min_air_temp_C', 13, 13],
                 33: ['normal_reanalysis_min_air_temp_C', 15, 15],
                 34: ['normal_reanalysis_min_air_temp_C', 17, 28],
                 25: ['station_min_temp_c', 0, 5],
                 26: ['station_min_temp_c', 12, 27],

                 # LN_station_precip_mm - weeks 10-13, 18
                 27: ['LN_station_precip_mm', 10, 13],
                 28: ['LN_station_precip_mm', 18, 18],

                 # normal_reanalysis_max_air_temp_C - weeks 6-14, 25-32
                 29: ['normal_reanalysis_max_air_temp_C', 6, 14],
                 30: ['normal_reanalysis_max_air_temp_C', 25, 32],

                 # normal_reanalysis_min_air_temp_C - weeks 0-7, 13, 15, 17-28
                 31: ['normal_reanalysis_min_air_temp_C', 0, 7],
                 32: ['normal_reanalysis_min_air_temp_C', 13, 13],
                 33: ['normal_reanalysis_min_air_temp_C', 15, 15],
                 34: ['normal_reanalysis_min_air_temp_C', 17, 28],

                 # normal_reanalysis_avg_temp_C - weeks 0-11, 23-32
                 35: ['normal_reanalysis_avg_temp_C', 0, 11],
                 36: ['normal_reanalysis_avg_temp_C', 23, 32],

                 # normal_reanalysis_tdtr_C - weeks 0-1, 10-20, 27-32
                 37: ['normal_reanalysis_tdtr_C', 0, 1],
                 38: ['normal_reanalysis_tdtr_C', 10, 20],
                 39: ['normal_reanalysis_tdtr_C', 27, 32]}
```

Figure 4.5B: Dictionary containing climatic_var, start_lag, end_lag - iq_train

```
def FeatureEng(df, dict_df) :
    """
    Objectives:
        1) Create lagged climatic variables as new features for df based on criteria in dict_df
        2) Add those newly created lagged climatic variables to df
        3) Return a list of new columns
    Args:
        df:        dataframe contains transformed variables
        dict_df:   dictionary contains continous lags for each variable
                    (break down discountinous lags into separate keys)
    Returns:
        df:              original dataframe plus new variables
        list(Agg_df):    list of new variables
    Raises:
        Nothing
    """
        # Create lagged variables (from nearest to furthest weeks)
        for i in range(value[1], value[2] + 1):

            # Create a column shift back by i week & append to cols
            cols.append(df6.shift(i))

            # Create corresponding column name & append to names
            if i == 0:
                title = 'FE_' + climate_var + '(t)'
                names += [title]
            else:
                title = 'FE_' + climate_var + '(t-%d)' % (i)
                names += [title]

    # put it all together
    Agg_df = pd.concat(cols, axis=1)
    Agg_df.columns = names

    print("Top 33 rows of new columns")
    print("\n")
    print(Agg_df.head(33))

    # Merge Agg_df with df
    df = pd.merge(df, Agg_df, on='City_week_start_date', how='inner')

    print("\nNumber of columns added from FeatureEng() = ", len(Agg_df.columns))
    print("\n")
    print(df.info(verbose = True, null_counts = True))

    # Return the list of new predictors
    return df, list(Agg_df)
```

Figure 4.5C: UDF for feature engineering

```
In [117]: # sj_train - Function call for FeatureEng(df, dict_df)
          # Feature engineering to create lagged climatic variables based on dict_sj_train
          # Return the list of new columns
          sj_train, list_FE_sj_train = FeatureEng(sj_train, dict_sj_train)

          Number of columns added from FeatureEng() =  364

          FE_normal_reanalysis_min_air_temp_C(t-29)        903 non-null float64
          FE_normal_reanalysis_min_air_temp_C(t-30)        902 non-null float64
          FE_normal_reanalysis_min_air_temp_C(t-31)        901 non-null float64
          FE_normal_reanalysis_min_air_temp_C(t-32)        900 non-null float64
          FE_normal_reanalysis_tdtr_C(t)                   932 non-null float64
          FE_normal_reanalysis_tdtr_C(t-1)                 931 non-null float64
          FE_normal_reanalysis_tdtr_C(t-2)                 930 non-null float64
          FE_normal_reanalysis_tdtr_C(t-3)                 929 non-null float64
          FE_normal_reanalysis_tdtr_C(t-4)                 928 non-null float64
          FE_normal_reanalysis_tdtr_C(t-5)                 927 non-null float64
          FE_normal_reanalysis_tdtr_C(t-6)                 926 non-null float64
          FE_normal_reanalysis_tdtr_C(t-7)                 925 non-null float64
          FE_normal_reanalysis_tdtr_C(t-8)                 924 non-null float64
          FE_normal_reanalysis_tdtr_C(t-30)                902 non-null float64
          FE_normal_reanalysis_tdtr_C(t-31)                901 non-null float64
          FE_normal_reanalysis_tdtr_C(t-32)                900 non-null float64
          dtypes: float64(385), int32(1), object(3)
          memory usage: 2.8+ MB
          None
```

Figure 4.5D: Partial list of 364 newly engineered features for sj_train

```
In [119]: # iq_train - Function call for FeatureEng(df, dict_df)
          # Feature engineering to create lagged climatic variables based on dict_sj_train
          # Return the list of new columns
          iq_train, list_FE_iq_train = FeatureEng(iq_train, dict_iq_train)

          Number of columns added from FeatureEng() =  241

          FE_normal_reanalysis_tdtr_C(t-11)                505 non-null float64
          FE_normal_reanalysis_tdtr_C(t-12)                504 non-null float64
          FE_normal_reanalysis_tdtr_C(t-13)                503 non-null float64
          FE_normal_reanalysis_tdtr_C(t-14)                502 non-null float64
          FE_normal_reanalysis_tdtr_C(t-15)                501 non-null float64
          FE_normal_reanalysis_tdtr_C(t-16)                500 non-null float64
          FE_normal_reanalysis_tdtr_C(t-17)                499 non-null float64
          FE_normal_reanalysis_tdtr_C(t-18)                498 non-null float64
          FE_normal_reanalysis_tdtr_C(t-19)                497 non-null float64
          FE_normal_reanalysis_tdtr_C(t-20)                496 non-null float64
          FE_normal_reanalysis_tdtr_C(t-27)                489 non-null float64
          FE_normal_reanalysis_tdtr_C(t-28)                488 non-null float64
          FE_normal_reanalysis_tdtr_C(t-29)                487 non-null float64
          FE_normal_reanalysis_tdtr_C(t-30)                486 non-null float64
          FE_normal_reanalysis_tdtr_C(t-31)                485 non-null float64
          FE_normal_reanalysis_tdtr_C(t-32)                484 non-null float64
          dtypes: float64(262), int32(1), object(3)
          memory usage: 1.0+ MB
          None
```

Figure 4.5E: Partial list of 241 newly engineered features for iq_train

364 and 241 new features were added to sj_train (Figure 4.5D) and iq_train (Figure 4.5E) respectively.

CHAPTER 5
RESULTS AND ANALYSIS

5.1 Four Lasso Models to Be Built

Table 5.1: Four Lasso Models

City	Base model with no feature engineering	Model with feature engineering from lagged climatic variables
San Juan	M1	M2
Iquitos	M3	M4

5.2 San Juan's M1 - Base Lasso Model with No Feature Engineering

Please refer to the following sections in attached Jupyter notebook:

- PART 5 - Train San Juan's Base Lasso Model
- PART 6 - Fit Train San Juan's Base Lasso Model Using Testing Dataset
- PART 7 - Undo Transformation To Obtain Predicted total_cases

5.2.1 Part 5A - Prepare Independent & Dependent Variables

The initial 32 weeks with NaN from lagged climatic variables were deleted in order to standardize between the base and feature engineering models.

```
# sj_train, Model M1 specification - most basic with no FE or ln for dependent variable
city_M1 = 'sj'
df_M1 = sj_train
AddFE_M1 = False
DependentVar_M1 = 'Advance4W_normal_total_cases'

# sj_train, Model M1 - Function call to Prepare_Xy(city, df, AddFE, DependentVar)
X_sj_train_M1, y_sj_train_M1, X_label_sj_train_M1 = Prepare_Xy(city_M1, df_M1, AddFE_M1, DependentVar_M1)

Number of rows in independent & dependent variables matched after removing NaN from 32 weeks lagged variables.
```

Figure 5.2.1: Prepare X & y as Lasso inputs

5.2.2 Part 5B - Prepare Training & Validation Datasets at Simple 70:30 split

The first 70% of X and y from PART 5A were assigned as subtrain dataset while the last 30% were assigned as subvalid datasets. According to Figure 5.2.2A, subtrain ranged from 10-Dec-1990 to 15-Jan-2003 while subvalid ranged from 22-Jan-2003 to 25-Mar-2008.

```
# sj_train, Model 1
# X_sj_train_M1, y_sj_train_M1 - Function call to TrainValidSplit(X, y)
X_sj_subtrain_M1, y_sj_subtrain_M1, X_sj_subvalid_M1, y_sj_subvalid_M1, sj_index_subtrain_M1, sj_index_subvalid_M1 = TrainValidS
```

```
SubTrain period - row indices:
Index(['sj1990-12-10', 'sj1990-12-17', 'sj1990-12-24', 'sj1991-01-01',
       'sj1991-01-08', 'sj1991-01-15', 'sj1991-01-22', 'sj1991-01-29',
       'sj1991-02-05', 'sj1991-02-12',
       ...
       'sj2002-11-12', 'sj2002-11-19', 'sj2002-11-26', 'sj2002-12-03',
       'sj2002-12-10', 'sj2002-12-17', 'sj2002-12-24', 'sj2003-01-01',
       'sj2003-01-08', 'sj2003-01-15'],
      dtype='object', name='City_week_start_date', length=630)

SubValid period - row indices:
Index(['sj2003-01-22', 'sj2003-01-29', 'sj2003-02-05', 'sj2003-02-12',
       'sj2003-02-19', 'sj2003-02-26', 'sj2003-03-05', 'sj2003-03-12',
       'sj2003-03-19', 'sj2003-03-26',
       ...
       'sj2008-01-22', 'sj2008-01-29', 'sj2008-02-05', 'sj2008-02-12',
       'sj2008-02-19', 'sj2008-02-26', 'sj2008-03-04', 'sj2008-03-11',
       'sj2008-03-18', 'sj2008-03-25'],
      dtype='object', name='City_week_start_date', length=270)
```

Figure 5.2.2A: Split X and y into subtrain and subvalid

5.2.3 Part 5C - Basic Lasso Using Data Split (70:30) & Alpha = 0.1

Figure 5.2.2B list the parameter setting of Lasso function.

```
def Lasso_Specification(Alpha1):
    """
        Standardize the parameters of Lasso function with only alpha value is flexible
        Set normalize = False if using standardized inputs
    """
    LassoModel = Lasso(alpha = Alpha1, fit_intercept=False, max_iter=200000, normalize=False, positive=False,
                  random_state=123, tol=0.001, warm_start=False, selection = 'random')

    return LassoModel
```

Figure 5.2.2B: Setting for Lasso function

An UDF named BasicLasso was written to run a basic Lasso model using a random alpha of 0.1 as Part of the progression toward grid search cross validation. According to Figure 5.2.2C, 5/15 of the predictors have non-zero coefficients.

```
# sj_subtrain, Model 1 - Function call for Basic_Lasso1(X_subtrain, y_subtrain, X_subvalid, y_subvalid, X_label)
Basic_Lasso1(X_sj_subtrain_M1, y_sj_subtrain_M1, X_sj_subvalid_M1, y_sj_subvalid_M1, X_label_sj_train_M1)
```

```
Alpha:  0.1
Subtrain MAE =  1.6738
Subvalid MAE =  1.4834
Time lapsed:  0.012 seconds
```

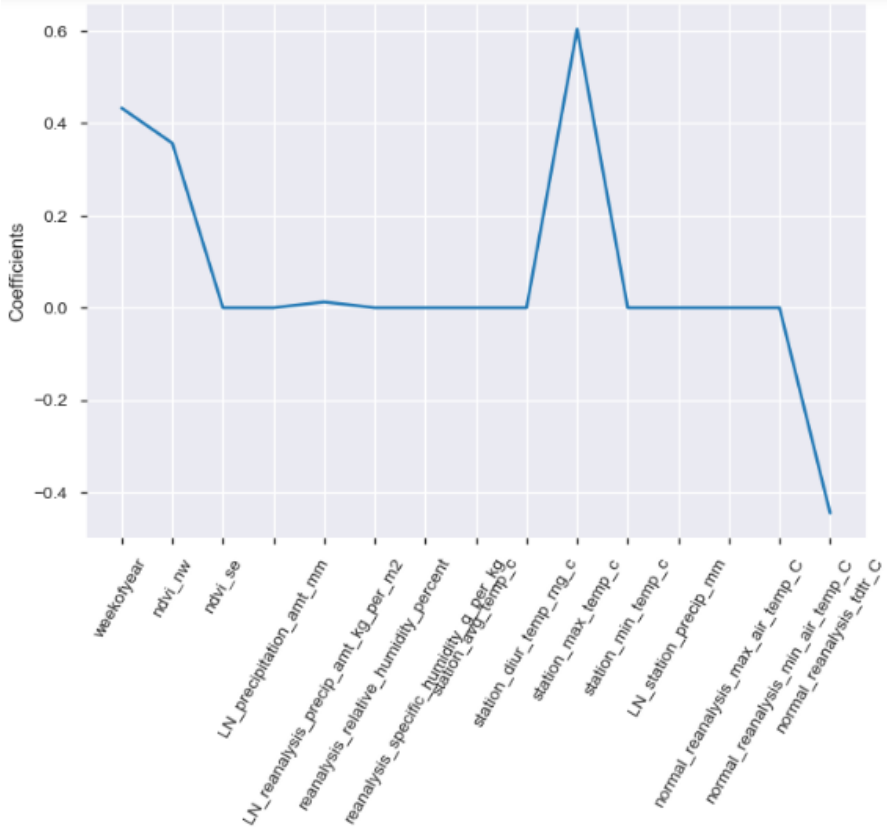


Figure 5.2.2C: M1 - BasicLasso for San Juan using alpha = 0.1

5.2.4 Part 5D - Basic Lasso (Simple 70:30 Data Split) Over Different Alphas

An UDF entitled BasicLasso2 extended check the impact of different alphas on predictors' coefficients. According to Lasso Coefficient Path in Figure 5.2.2D,

- San Juan's initial coefficients ranges from (-1.25 to +0.75)
- The magnitude of coefficients decrease as alpha increases
- All coefficients terrorized as alpha approaches 1.

```
# Set alpha range
Alphas2_M1 = np.logspace(-2, 1, 50)

# sj_train, Model 1 - Function call to Basic_Lasso2(Alpha2, X_subtrain, y_subtrain)
Basic_Lasso2(Alphas2_M1, X_sj_subtrain_M1, y_sj_subtrain_M1)
```

```
Time lapsed:  0.0735 seconds
```

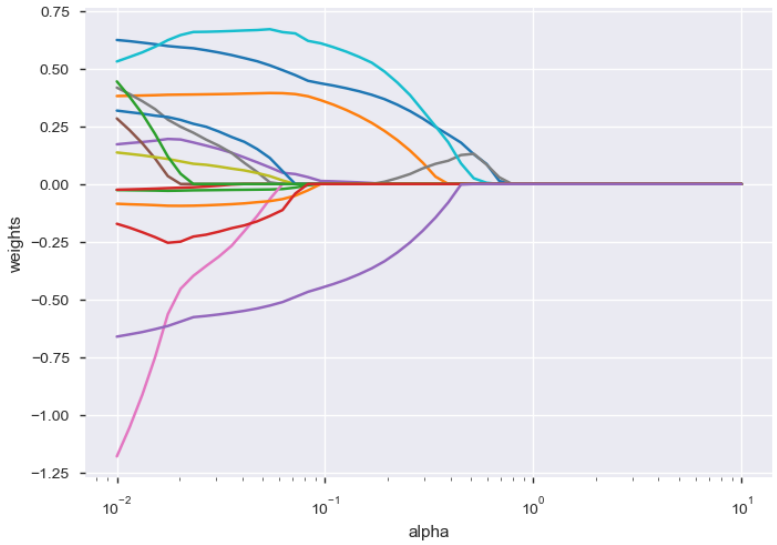


Figure 5.2.4: M1 - Lasso Coefficient Path for San Juan

5.2.5 Part 5E - Optimize Lasso Using GridSearchCV with 10-splits from TimeSeriesSplit

An UDF entitled Lasso_GridSearchCV_TimeSeriesSplit perform GridSearchCV with 10-splits from TimeSeriesSplit to choose the best alpha, followed by refitting the model, and compute the associated subvalid error. According to Figure 5.2.5A, optimal alpha = 1.072267222010323, where the negative MAE chart peak. However, this corresponds to weight = 0 in Figure 5.2.4. This will cause model with insignificant coefficients and zero predictions which are not useful at all.

```
Alphas3_M1_init = np.logspace(-2, 1, 100)

# sj_train, Model 1 - Function call to Lasso_GridSearchCV_TimeSeriesSplit(Alphas3, X_subtrain, y_subtrain, X_subvalid, y_subvali
Lasso_sj_M1_init = Lasso_GridSearchCV_TimeSeriesSplit(Alphas3_M1_init, X_sj_subtrain_M1, y_sj_subtrain_M1,
                                                      X_sj_subvalid_M1, y_sj_subvalid_M1)
```

```
Optimized alpha:  1.072267222010323
Subtrain MAE =  1.7231
Subvalid MAE =  0.9185
Time lapsed:  1.7733  seconds
```

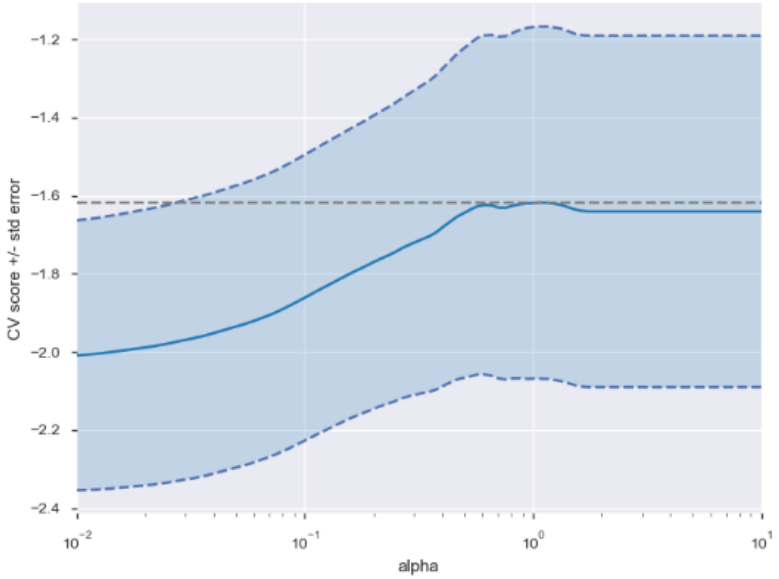


Figure 5.2.5A: M1 - initial grid search for San Juan

Therefore, we need to fine tune the alphas to be the range of 0.01-0.1 where there are at least a handful of non-zero weights. According to Figure 5.2.5B, an unique solution was found at alpha = 0.1 and MAE improved from subtrain 1.3401 to subvalid 1.1104.

```
# Choose alpha range where there are at least a handful of predictors will remain
Alphas3_M1 = np.logspace(-2, -1, 100)

# sj_train, Model 1 - Function call to Lasso_GridSearchCV_TimeSeriesSplit(Alphas3, X_subtrain, y_subtrain, X_subvalid, y_subvalid
Lasso_sj_M1 = Lasso_GridSearchCV_TimeSeriesSplit(Alphas3_M1, X_sj_subtrain_M1, y_sj_subtrain_M1,
                                                 X_sj_subvalid_M1, y_sj_subvalid_M1)
```

```
Optimized alpha:  0.1
Subtrain MAE =  1.6738
Subvalid MAE =  1.4834
Time lapsed:  2.613  seconds
```

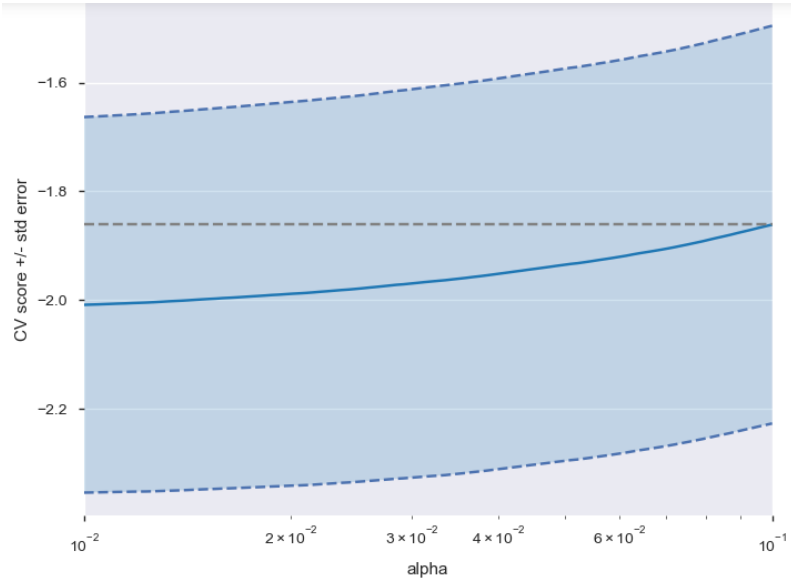


Figure 5.2.5B: M1 - refined grid search for San Juan

An UDF entitled Lasso_Coef map Lasso coefficients with predictors' names and plot a bar chart of predictors in descending order of absolute coefficients. According to Figure 5.2.5C, 5/15 predictors were found to be significant in predicting Advance4W_normal_total_cases with non-zero coefficients. They are

- station_max_temp_C - the hotter, the more cases.
- normal_reanalysis_tdtr_C - the bigger the gap between maximum & minimum temperatures, the lesser cases.
- weekofyear - cases increase toward year end.
- ndvi_nw - the more vegetation on the north-west direction, the more cases.
- LN_renalaysis_precip_amt_kg_per_m2 - the more rainfall, the more cases

```
# sj_train, Model 1
# Lasso_sj_M1, X_sj_subtrain_M1 - Function call to Lasso_Coef(model, X_subtrain, X_label)
Lasso_Coef_sj_M1 = Lasso_Coef(Lasso_sj_M1, X_sj_subtrain_M1, X_label_sj_train_M1)
```

```
Non-zero coefficient in descending absolute value order:
                                    Coef
station_max_temp_c                 0.604519
normal_reanalysis_tdtr_C          -0.445335
weekofyear                         0.433175
ndvi_nw                            0.357090
LN_reanalysis_precip_amt_kg_per_m2  0.012586
```

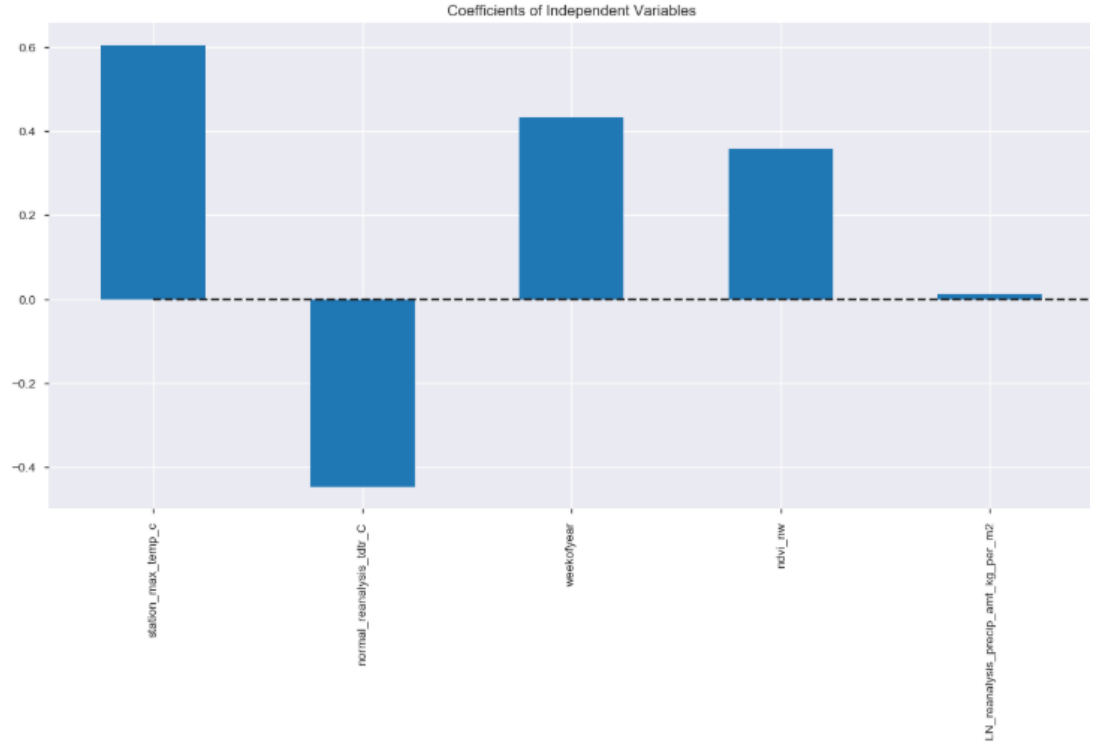


Figure 5.2.5C: M1 - bar chart of predictors in descending order of absolute coefficients

5.2.6 Part 5F - Create Dataframe for Lasso Predictions

The fitted values from Lasso function are in matrix form without row index. An UDF entitled LassoPredict map the predictions with the City_week_start_date unique identifier. Figures 5.2.6 A-B Partially list the predicted Advance4W_normal_total_cases of San Juan subtrain and subvalid using M1 model.

```
# sj_train, Model 1 - Function call to LassoPredict(Lasso_model, X, row_index, DependentVar)
Predict_sj_subtrain_M1 = LassoPredict(Lasso_sj_M1, X_sj_subtrain_M1, sj_index_subtrain_M1, DependentVar_M1)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 630 entries, sj1990-12-10 to sj2003-01-15
Data columns (total 1 columns):
Advance4W_normal_total_cases    630 non-null float64
dtypes: float64(1)
memory usage: 9.8+ KB
None
                     Advance4W_normal_total_cases
City_week_start_date
sj1990-12-10                             0.218226
sj1990-12-17                             0.817412
sj1990-12-24                             0.793128
sj1991-01-01                             0.000000
sj1991-01-08                             0.000000
                     Advance4W_normal_total_cases
City_week_start_date
sj2002-12-17                                  0.0
sj2002-12-24                                  0.0
sj2003-01-01                                  0.0
sj2003-01-08                                  0.0
sj2003-01-15                                  0.0
```

Figure 5.2.6A: M1 - Predicted Advance4W_normal_total_cases for San Juan subtrain

```
# sj_train, Model 1 - Function call to LassoPredict(Lasso_model, X, row_index, DependentVar)
Predict_sj_subvalid_M1 = LassoPredict(Lasso_sj_M1, X_sj_subvalid_M1, sj_index_subvalid_M1, DependentVar_M1)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 270 entries, sj2003-01-22 to sj2008-03-25
Data columns (total 1 columns):
Advance4W_normal_total_cases    270 non-null float64
dtypes: float64(1)
memory usage: 4.2+ KB
None
                     Advance4W_normal_total_cases
City_week_start_date
sj2003-01-22                                  0.0
sj2003-01-29                                  0.0
sj2003-02-05                                  0.0
sj2003-02-12                                  0.0
sj2003-02-19                                  0.0
                     Advance4W_normal_total_cases
City_week_start_date
sj2008-02-26                                  0.0
sj2008-03-04                                  0.0
sj2008-03-11                                  0.0
```

Figure 5.2.6B: M1 - Predicted Advance4W_normal_total_cases for San Juan subvalid

5.2.7 Part 6A - Preprocess sj_test

The test dataset of San Juan was preprocessed via

- Figure 5.2.7A - rescaling
- Figure 5.2.7B - single imputation of MICE
- Figure 5.2.7C - removal of highly correlated variables
- Figure 5.2.7D - log transformation of right-skewed variables
- Figure 5.2.7E - standardization of numerical variables

```
# PART 2F - Rescaling of variables (temperatures)
# sj_test - Function call to rescaling(df)
rescaling(sj_test)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 264 entries, sj2008-04-01 to sj2013-04-23
Data columns (total 29 columns):
year                                    264 non-null int64
weekofyear                              264 non-null int64
week_start_date                         264 non-null object
ndvi_ne                                 221 non-null float64
ndvi_nw                                 252 non-null float64
ndvi_se                                 263 non-null float64
ndvi_sw                                 263 non-null float64
precipitation_amt_mm                    262 non-null float64
reanalysis_precip_amt_kg_per_m2         262 non-null float64
reanalysis_relative_humidity_percent    262 non-null float64
reanalysis_sat_precip_amt_mm            262 non-null float64
reanalysis_specific_humidity_g_per_kg   262 non-null float64
station_avg_temp_c                      262 non-null float64
station_diur_temp_rng_c                 262 non-null float64
station_max_temp_c                      262 non-null float64
station_min_temp_c                      262 non-null float64
```

Figure 5.2.7A: San Juan test dataset - rescaling

```
# PART 3C - Missing Values Analysis & Treatment
# sj_test - Function call to MICE_imputation(df, list_variables)
MICE_imputation(sj_test, list_test_variables)

# Important that                      - Advance4W_normal_total_cases            0 non-null float64
# Correct since these ar the unlaggeds - total_cases                            4 non-null float64
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 264 entries, sj2008-04-01 to sj2013-04-23
Data columns (total 29 columns):
year                                    264 non-null float64
weekofyear                              264 non-null float64
week_start_date                         264 non-null object
ndvi_ne                                 264 non-null float64
ndvi_nw                                 264 non-null float64
ndvi_se                                 264 non-null float64
ndvi_sw                                 264 non-null float64
precipitation_amt_mm                    264 non-null float64
reanalysis_precip_amt_kg_per_m2         264 non-null float64
reanalysis_relative_humidity_percent    264 non-null float64
reanalysis_sat_precip_amt_mm            264 non-null float64
reanalysis_specific_humidity_g_per_kg   264 non-null float64
station_avg_temp_c                      264 non-null float64
station_diur_temp_rng_c                 264 non-null float64
station_max_temp_c                      264 non-null float64
station_min_temp_c                      264 non-null float64
station_precip_mm                       264 non-null float64
total_cases                             4 non-null float64
dataset_label                           264 non-null object
Population_inter                        264 non-null int32
city                                    264 non-null object
```

Figure 5.2.7B: San Juan test dataset - rescaling single imputation of MICE

```
# PART 4A - Feature Selection (Round 1 - Based on EDA)
# Drop the same variables as per sj_train
sj_test.drop(list_sj_train_drop, axis=1, inplace = True)

sj_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 264 entries, sj2008-04-01 to sj2013-04-23
Data columns (total 21 columns):
weekofyear                              264 non-null float64
week_start_date                         264 non-null object
ndvi_nw                                 264 non-null float64
ndvi_se                                 264 non-null float64
precipitation_amt_mm                    264 non-null float64
reanalysis_precip_amt_kg_per_m2         264 non-null float64
reanalysis_relative_humidity_percent    264 non-null float64
reanalysis_specific_humidity_g_per_kg   264 non-null float64
station_avg_temp_c                      264 non-null float64
station_diur_temp_rng_c                 264 non-null float64
station_max_temp_c                      264 non-null float64
station_min_temp_c                      264 non-null float64
station_precip_mm                       264 non-null float64
total_cases                             4 non-null float64
dataset_label                           264 non-null object
Population_inter                        264 non-null int32
city                                    264 non-null object
Advance4W_normal_total_cases            0 non-null float64
normal_reanalysis_max_air_temp_C        264 non-null float64
normal_reanalysis_min_air_temp_C        264 non-null float64
normal_reanalysis_tdtr_C                264 non-null float64
```

Figure 5.2.7C: San Juan test dataset – removal of highly correlated variables

```
# Function call for LogTransformation_sj(df, TestBoolean)
# Transform right-skewed distributions in sj_train using ln(1+x)
LogTransformation_sj(sj_train, TestBoolean = False)
```

```
station_avg_temp_c                      932 non-null float64
station_diur_temp_rng_c                 932 non-null float64
station_max_temp_c                      932 non-null float64
station_min_temp_c                      932 non-null float64
station_precip_mm                       932 non-null float64
total_cases                             932 non-null float64
dataset_label                           932 non-null object
Population_inter                        932 non-null int32
city                                    932 non-null object
Advance4W_normal_total_cases            932 non-null float64
normal_reanalysis_max_air_temp_C        932 non-null float64
normal_reanalysis_min_air_temp_C        932 non-null float64
normal_reanalysis_tdtr_C                932 non-null float64
LN_precipitation_amt_mm                 932 non-null float64
LN_reanalysis_precip_amt_kg_per_m2      932 non-null float64
LN_station_precip_mm                    932 non-null float64
LN_Advance4W_normal_total_cases         932 non-null float64
dtypes: float64(21), int32(1), object(3)
memory usage: 185.7+ KB
```

Figure 5.2.7D: San Juan test dataset – log transformation of right-skewed variables

```
# PART 4C - Standardization Transformation
# sj_test - Function call to Standardization(df, list_df) using the same list_standardize_sj
sj_test_Standard = Standardization(sj_test, list_shortlist_sj)

# Descriptive statistics, round to 2 decimal places
# The output should have zero means & 1 standard deviations
round(sj_test_Standard.describe(), 2)

sj_test = sj_test_Standard
```

Figure 5.2.7E: San Juan test dataset – standardization of numerical variables

5.2.8 Part 6B - Feature Engineering for sj_test

An UDF entitled Lengthen_Test add bottom 32 rows from sj_train to be on top of sj_test in order to create 32 weeks lags (Figure 5.2.8 A-B).

```
# sj - Function call to Lengthen_Test(df_train, df_test, city)
sj_test = Lengthen_Test(sj_train, sj_test, 'sj')
```

```
Lengthened sj_test:


<class 'pandas.core.frame.DataFrame'>
Index: 296 entries, sj2007-08-20 to sj2013-04-23
Data columns (total 25 columns):
weekofyear                              296 non-null float64
week_start_date                         296 non-null object
ndvi_nw                                 296 non-null float64
ndvi_se                                 296 non-null float64
precipitation_amt_mm                    296 non-null float64
reanalysis_precip_amt_kg_per_m2         296 non-null float64
reanalysis_relative_humidity_percent    296 non-null float64
reanalysis_specific_humidity_g_per_kg   296 non-null float64
station_avg_temp_c                      296 non-null float64
station_diur_temp_rng_c                 296 non-null float64
station_max_temp_c                      296 non-null float64
station_min_temp_c                      296 non-null float64
```

Figure 5.2.8A: San Juan test dataset – add bottom 32 weeks from sj_train

```
# PART 4E - Feature Engineering Based on CCF Analysis in R
# sj_test - Function call for FeatureEng(df, dict_df)
sj_test, list_FE_sj_train = FeatureEng(sj_test, dict_sj_train)
```

```
Top 33 rows of new columns
```

City_week_start_date	FE_ndvi_nw(t)	FE_ndvi_nw(t-1)	FE_ndvi_nw(t-2)
sj2007-08-20	0.274676	NaN	NaN
sj2007-08-27	-0.044557	0.274676	NaN
sj2007-09-03	-0.270373	-0.044557	0.274676
sj2007-09-10	-0.120319	-0.270373	-0.044557
sj2007-09-17	0.078281	-0.120319	-0.270373
sj2007-09-24	-0.780556	0.078281	-0.120319
sj2007-10-01	-0.474858	-0.780556	0.078281
sj2007-10-08	0.399353	-0.474858	-0.780556
sj2007-10-15	0.715643	0.399353	-0.474858
sj2007-10-22	-0.849074	0.715643	0.399353
sj2007-10-29	-0.149006	-0.849074	0.715643
sj2007-11-05	-1.284340	-0.149006	-0.849074
sj2007-11-12	-1.438256	-1.284340	-0.149006
sj2007-11-19	-0.899542	-1.438256	-1.284340

Figure 5.2.8B: San Juan test dataset – feature engineering

X and y preparation was conducted (Figure 5.2.8C) before generating prediction using M1 model (Figure 5.2.8D).

```
# Part 5A - Prepare Independent & Dependent Variables
# sj_train, Model M1 - Function call to Prepare_Xy(city, df, AddFE, DependentVar)
X_sj_test_M1, y_sj_test_M1, X_label_sj_test_M1 = Prepare_Xy(city_M1, sj_test, AddFE_M1, DependentVar_M1)
```

```
Number of rows in independent & dependent variables matched after removing NaN from 32 weeks lagged variables.


<class 'pandas.core.frame.DataFrame'>
Index: 264 entries, sj2008-04-01 to sj2013-04-23
Data columns (total 15 columns):
weekofyear                              264 non-null float64
ndvi_nw                                 264 non-null float64
ndvi_se                                 264 non-null float64
LN_precipitation_amt_mm                 264 non-null float64
LN_reanalysis_precip_amt_kg_per_m2      264 non-null float64
reanalysis_relative_humidity_percent    264 non-null float64
reanalysis_specific_humidity_g_per_kg   264 non-null float64
station_avg_temp_c                      264 non-null float64
station_diur_temp_rng_c                 264 non-null float64
station_max_temp_c                      264 non-null float64
station_min_temp_c                      264 non-null float64
LN_station_precip_mm                    264 non-null float64
normal_reanalysis_max_air_temp_C        264 non-null float64
normal_reanalysis_min_air_temp_C        264 non-null float64
normal_reanalysis_tdtr_C                264 non-null float64
dtypes: float64(15)
memory usage: 33.0+ KB
```

Figure 5.2.8C: San Juan test dataset – prepare X and y

```
# PART 5F - Create Dataframe for Lasso Predictions
# Model 1, X_sj_test - Function call to LassoPredict(Lasso_model, X, row_index)
Predict_sj_test_M1 = LassoPredict(Lasso_sj_M1, X_sj_test_M1, sj_index_test, DependentVar_M1)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 264 entries, sj2008-04-01 to sj2013-04-23
Data columns (total 1 columns):
Advance4W_normal_total_cases    264 non-null float64
dtypes: float64(1)
memory usage: 4.1+ KB
None
                      Advance4W_normal_total_cases
City_week_start_date
sj2008-04-01                                   0.0
sj2008-04-08                                   0.0
sj2008-04-15                                   0.0
sj2008-04-22                                   0.0
sj2008-04-29                                   0.0
                      Advance4W_normal_total_cases
City_week_start_date
sj2013-03-26                                   0.0
sj2013-04-02                                   0.0
sj2013-04-09                                   0.0
sj2013-04-16                                   0.0
sj2013-04-23                                   0.0
```

Figure 5.2.8D: San Juan test dataset – prediction from M1 model

5.2.9 Part 7A - Concatenate Lasso Predictions From y_sj_subtrain_M1, y_sj_subvalid_M1 & y_sj_test_M1

An UDF entitled Merge_LassoPred merged the predictions from predictions from subtrain, subvalid & test datasets.

```
# sj, Model 1 - Function call to Merge_LassoPred (Pred_subtrain, Pred _subvalid, Pred _test, Model_number)
Lasso_prediction_M1 = Merge_LassoPred(Predict_sj_subtrain_M1, Predict_sj_subvalid_M1, Predict_sj_test_M1, 'M1')
```

```
                      Advance4W_normal_Lasso_pred_M1
City_week_start_date
sj1990-12-10                                0.218226
sj1990-12-17                                0.817412
sj1990-12-24                                0.793128
sj1991-01-01                                0.000000
sj1991-01-08                                0.000000


                      Advance4W_normal_Lasso_pred_M1
City_week_start_date
sj2013-03-26                                     0.0
sj2013-04-02                                     0.0
sj2013-04-09                                     0.0
sj2013-04-16                                     0.0
sj2013-04-23                                     0.0
```

Figure 5.2.9: Merging the predictions from M1 model

5.2.10 Part 7B - Merge Lasso_prediction_M1 with sj_complete

An UDF entitled Merge_BasePredict_CityBase merge M1 model predictions with sj_complete which contains the actual total_cases.

```
# sj, Model 1 - Function call to Merge_BasePredict_CityBase(df_BasePredict, df_city)
sj_complete_M1 = Merge_BasePredict_CityBase(sj, Lasso_prediction_M1)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1196 entries, sj1990-04-30 to sj2013-04-23
Data columns (total 30 columns):
Advance4W_normal_Lasso_pred_M1          1164 non-null float64
year                                    1196 non-null int64
weekofyear                              1196 non-null int64
week_start_date                         1196 non-null object
ndvi_ne                                 962 non-null float64
ndvi_nw                                 1136 non-null float64
ndvi_se                                 1176 non-null float64
ndvi_sw                                 1176 non-null float64
precipitation_amt_mm                    1185 non-null float64
reanalysis_air_temp_k                   1188 non-null float64
reanalysis_avg_temp_k                   1188 non-null float64
reanalysis_dew_point_temp_k             1188 non-null float64
reanalysis_max_air_temp_k               1188 non-null float64
reanalysis_min_air_temp_k               1188 non-null float64
reanalysis_precip_amt_kg_per_m2         1188 non-null float64
reanalysis_relative_humidity_percent    1188 non-null float64
```

Figure 5.2.10: Merging the predictions from M1 model actual total_cases

5.2.11 Part 7D - Plot Actual vs. Predicted total_cases on the same chart

An UDF entitled Plot_Actual_Predict plot predicted and actual total_cases on the same chart. According to Figure 5.2.11,

- The predictions shows annual seasonality with peaks in year end but failed to replicate the outbreaks in train dataset.
- The test period is expected to have outbreak in end of 2008.

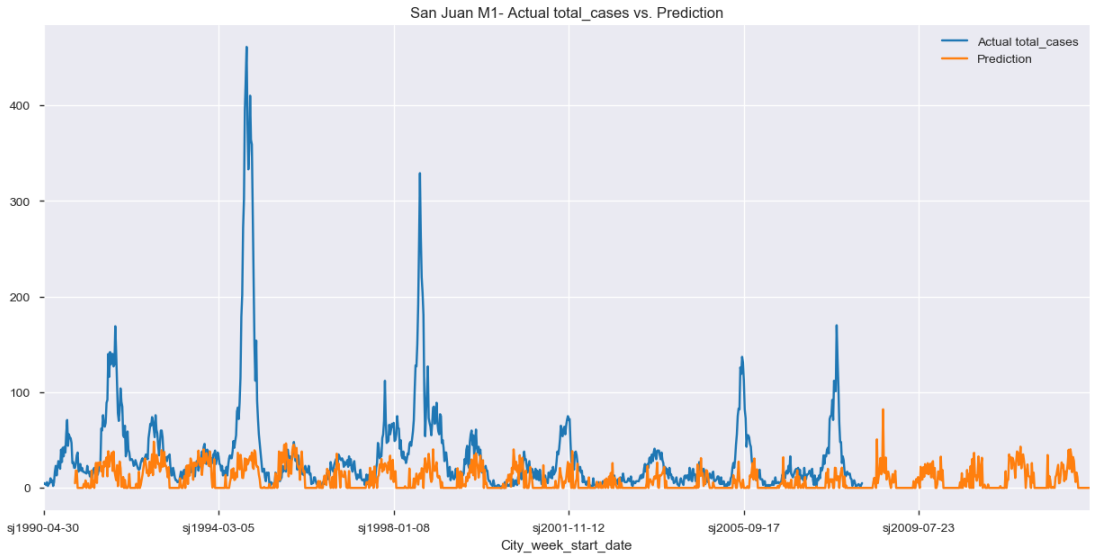


Figure 5.2.11: M1 model – predicted vs. actual total_cases

5.3 San Juan's M2 - Lasso Model with Feature Engineering from Lagged Climatic Variables

Please refer to "PART 8 - San Juan's Featured Engineered Lasso Model" in attached Jupyter notebook.

An UDF entitled Merge_TrainTestFE merged sj_train and sj_test (Figure 5.3A) before preparing X and y (Figure 5.3B) , followed by the splitting the dataset with feature engineering into subtrain vs. subvalid datasets (Figure 5.3C).

```
# sj - Function call to Merge_TrainTestFE(df_train, df_test)
sj_completeFE = Merge_TrainTestFE(sj_train, sj_test)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1228 entries, sj1990-04-30 to sj2013-04-23
Data columns (total 389 columns):
weekofyear                                  1228 non-null float64
week_start_date                             1228 non-null object
ndvi_nw                                     1228 non-null float64
ndvi_se                                     1228 non-null float64
precipitation_amt_mm                        1228 non-null float64
reanalysis_precip_amt_kg_per_m2             1228 non-null float64
reanalysis_relative_humidity_percent        1228 non-null float64
reanalysis_specific_humidity_g_per_kg       1228 non-null float64
station_avg_temp_c                          1228 non-null float64
station_diur_temp_rng_c                     1228 non-null float64
station_max_temp_c                          1228 non-null float64
station_min_temp_c                          1228 non-null float64
station_precip_mm                           1228 non-null float64
total_cases                                 968 non-null float64
dataset_label                               1228 non-null object
Population_inter                            1228 non-null int32
```

Figure 5.3A: M2 model – merger of sj_train and sj_test

```
# Part 5A - Prepare Independent & Dependent Variables

# Model M2 specification - most basic with no FE or ln for dependent variable
city_M2 = 'sj'
df_M2 = sj_train
AddFE_M2 = True
DependentVar_M2 = 'Advance4W_normal_total_cases'

# Model M1 - Function call to Prepare_Xy(city, df, AddFE, DependentVar)
X_sj_train_M2, y_sj_train_M2, X_label_sj_train_M2 = Prepare_Xy(city_M2, df_M2, AddFE_M2, DependentVar_M2)
```

```
Number of rows in independent & dependent variables matched after removing NaN from 32 weeks lagged variables.


<class 'pandas.core.frame.DataFrame'>
Index: 900 entries, sj1990-12-10 to sj2008-03-25
Data columns (total 365 columns):
weekofyear                                  900 non-null float64
FE_ndvi_nw(t)                               900 non-null float64
FE_ndvi_nw(t-1)                             900 non-null float64
FE_ndvi_nw(t-2)                             900 non-null float64
FE_ndvi_nw(t-3)                             900 non-null float64
FE_ndvi_nw(t-4)                             900 non-null float64
FE_ndvi_nw(t-5)                             900 non-null float64
FE_ndvi_nw(t-6)                             900 non-null float64
FE_ndvi_nw(t-7)                             900 non-null float64
FE_ndvi_nw(t-8)                             900 non-null float64
FE_ndvi_nw(t-9)                             900 non-null float64
FE_ndvi_nw(t-10)                            900 non-null float64
```

Figure 5.3B: M2 model – prepare X and y

```
# Part 5B - Prepare Training & Validation Datasets at Simple 70:30 split

# X_sj_train_M2 y_sj_train_M2- Function call to TrainValidSplit(X, y)
X_sj_subtrain_M2, y_sj_subtrain_M2, X_sj_subvalid_M2, y_sj_subvalid_M2, sj_index_subtrain_M2, sj_index_subvalid_M2 = TrainValidS
```

```
SubTrain period - row indices:
Index(['sj1990-12-10', 'sj1990-12-17', 'sj1990-12-24', 'sj1991-01-01',
       'sj1991-01-08', 'sj1991-01-15', 'sj1991-01-22', 'sj1991-01-29',
       'sj1991-02-05', 'sj1991-02-12',
       ...
       'sj2002-11-12', 'sj2002-11-19', 'sj2002-11-26', 'sj2002-12-03',
       'sj2002-12-10', 'sj2002-12-17', 'sj2002-12-24', 'sj2003-01-01',
       'sj2003-01-08', 'sj2003-01-15'],
      dtype='object', name='City_week_start_date', length=630)

SubValid period - row indices:
Index(['sj2003-01-22', 'sj2003-01-29', 'sj2003-02-05', 'sj2003-02-12',
       'sj2003-02-19', 'sj2003-02-26', 'sj2003-03-05', 'sj2003-03-12',
       'sj2003-03-19', 'sj2003-03-26',
       ...
       'sj2008-01-22', 'sj2008-01-29', 'sj2008-02-05', 'sj2008-02-12',
       'sj2008-02-19', 'sj2008-02-26', 'sj2008-03-04', 'sj2008-03-11',
       'sj2008-03-18', 'sj2008-03-25'],
      dtype='object', name='City_week_start_date', length=270)
```

Figure 5.3C: M2 model – subtrain and subvalid data split

According to Figure 5.3D, many coefficients were zerolised in Basic_Lasso1 fitting using alpha of 0.1.

```
# Part 5C - Basic Lasso Using Data Split (70:30) & Alpha = 0.1

# Model 2 - Function call for Basic_Lasso1(X_subtrain, y_subtrain, X_subvalid, y_subvalid, X_label)
Basic_Lasso1(X_sj_subtrain_M2, y_sj_subtrain_M2, X_sj_subvalid_M2, y_sj_subvalid_M2, X_label_sj_train_M2)
```

```
Alpha:  0.1
Subtrain MAE =  1.4532
Subvalid MAE =  2.2327
Time lapsed:  0.02 seconds
```

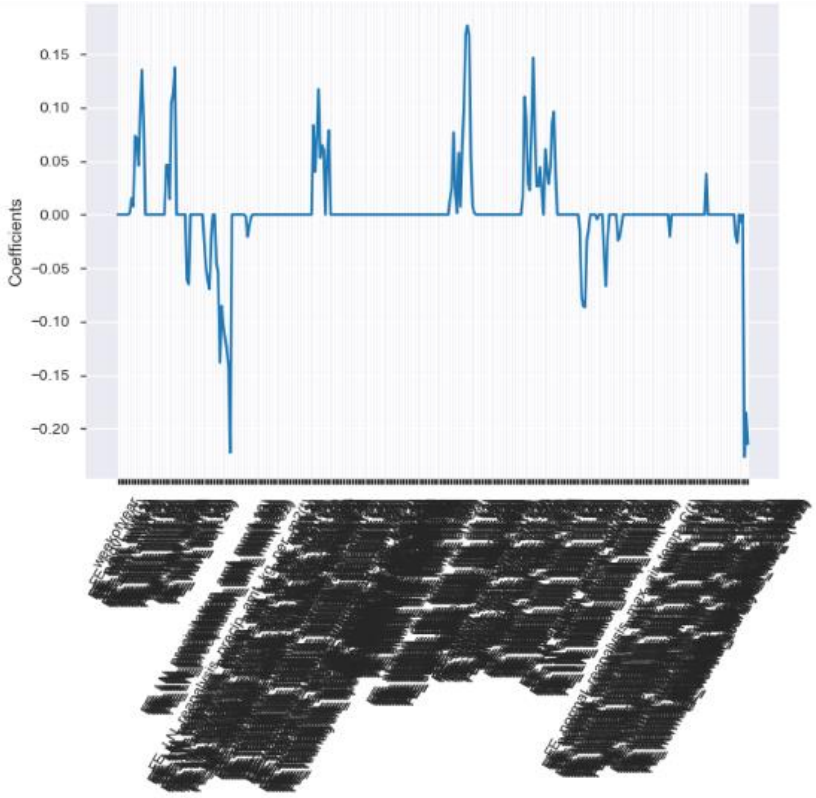


Figure 5.3D: M2 model – LassoBasic1

The Lasso coefficient path in Figure 5.3E demonstrated that

- Initial coefficients ranges from (-0.5 to +0.7).
- The magnitude of coefficients decrease as alpha increases.
- All coefficients zerrolized when alpha is larger than 1

```
# Part 5D - Basic Lasso (Simple 70:30 Data Split) Over Different Alphas

# Set alpha range
Alphas2_M2 = np.logspace(-2, 1, 50)

# sj_train, Model 2 - Function call to Basic_Lasso2(Alpha2, X_subtrain, y_subtrain)
Basic_Lasso2(Alphas2_M2, X_sj_subtrain_M2, y_sj_subtrain_M2)
```

```
Time lapsed:  1.1913 seconds
```

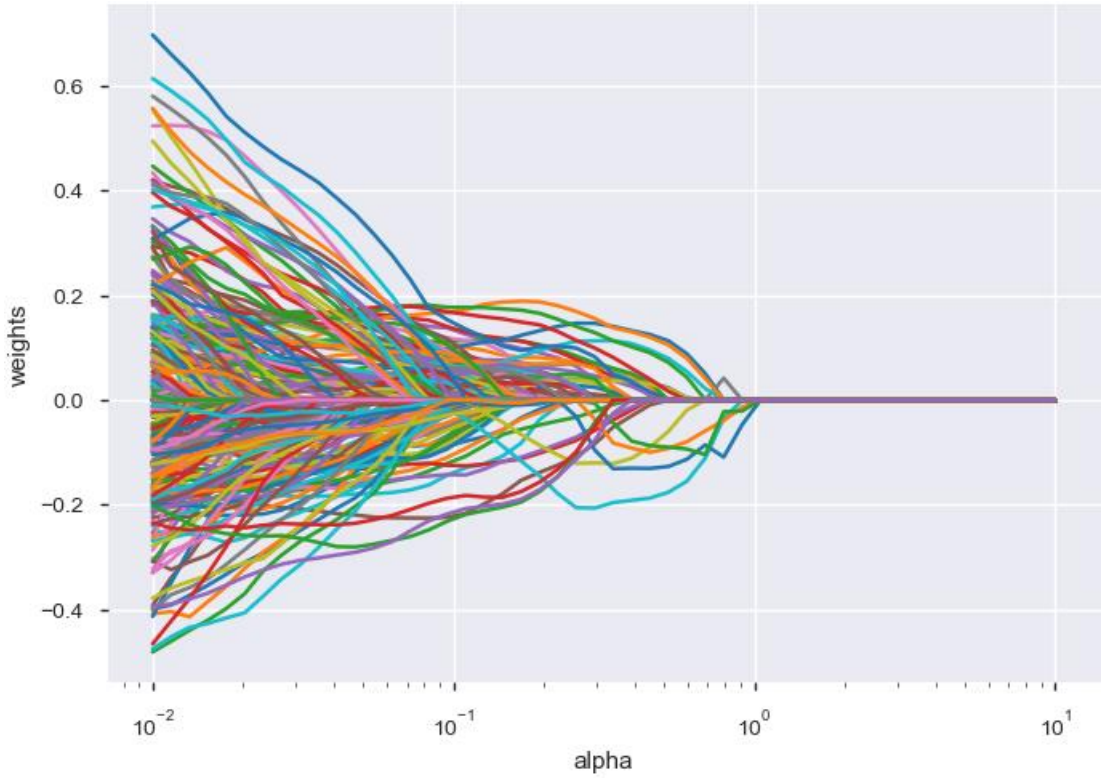


Figure 5.3E: M2 model – Lasso coefficient path

Initial grid search in Figure 5.3F found the optimal alpha = 1.3894954943731375, where the negative MAE chart peak. However, this corresponds to weight = 0 in Figure 5.3E. This will cause model with insignificant coefficients and zero predictions which are not useful at all.

```
# Part 5E - Optimize Lasso Using GridSearchCV with 10-splits from TimeSeriesSplit
Alphas3_M2_init = np.logspace(-2, 1, 50)

# sj_train, Model 2 - Function call to Lasso_GridSearchCV_TimeSeriesSplit(Alphas3, X_subtrain, y_subtrain, X_subvalid, y_subvali
Lasso_sj_M2_init = Lasso_GridSearchCV_TimeSeriesSplit(Alphas3_M2_init, X_sj_subtrain_M2, y_sj_subtrain_M2,
                                                      X_sj_subvalid_M2, y_sj_subvalid_M2)
```

```
Optimized alpha:  1.3894954943731375
Subtrain MAE =  1.7231
Subvalid MAE =  0.9185
Time lapsed:  11.2811  seconds
```

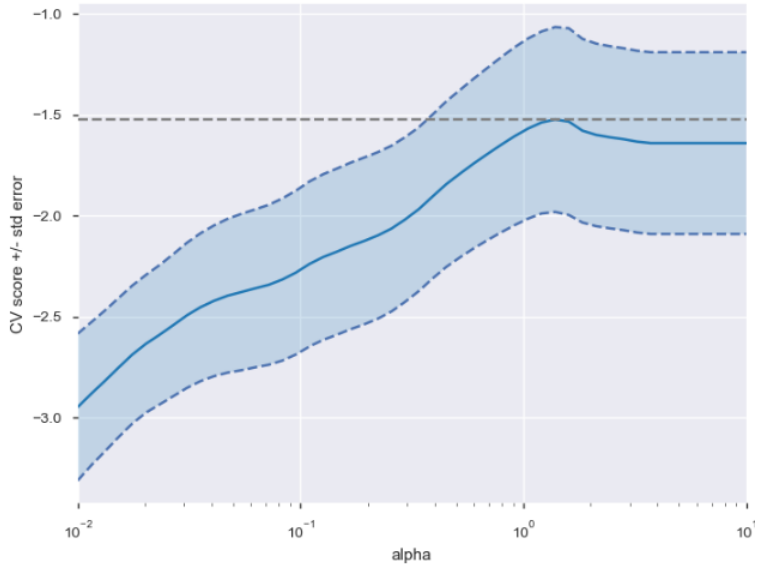


Figure 5.3F: M2 model – Initial grid search

Therefore, we need to fine tune the alphas to be the range of 10E-0.75 to 10E-0.35 where there are at least 5 sets of engineered features with non-zero weights. According to Figure 5.3G, unique solution was found at alpha = 0.5623413251903491 and MAE improved from subtrain 1.6195 to subvalid 1.1781.

```
# Choose alpha range where there are about 20 predictors will remain
Alphas3_M2 = np.logspace(-0.75, -0.35, 100)

# sj_train, Model 2 - Function call to Lasso_GridSearchCV_TimeSeriesSplit(Alphas3, X_subtrain, y_subtrain, X_subvalid, y_subvali
Lasso_sj_M2 = Lasso_GridSearchCV_TimeSeriesSplit(Alphas3_M2, X_sj_subtrain_M2, y_sj_subtrain_M2,
                                                 X_sj_subvalid_M2, y_sj_subvalid_M2)
```

```
Optimized alpha:  0.44668359215096315
Subtrain MAE =  1.583
Subvalid MAE =  1.3143
Time lapsed:  12.262  seconds
```

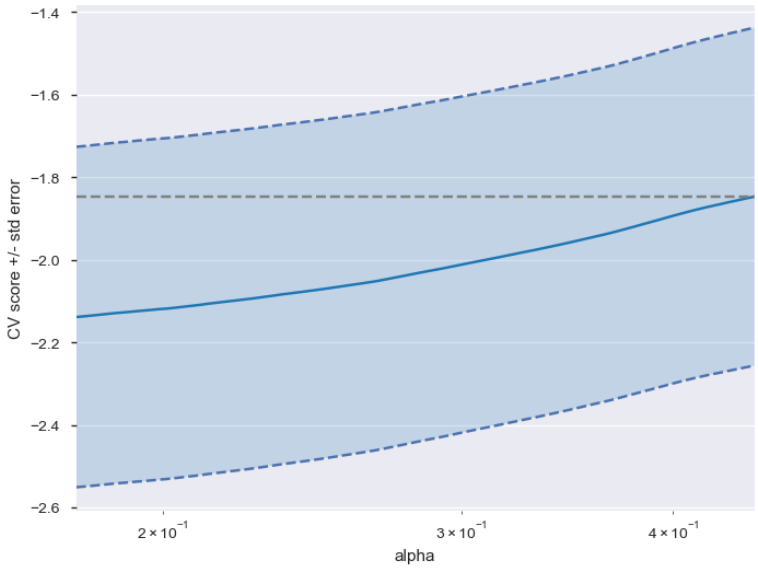


Figure 5.3G: M2 model – Refined grid search

According to Figure 5.3H, 19/365 predictors were found to be significant in predicting Advance4W_normal_total_cases with non-zero coefficients. The top 5 significant predictors are

- FE_normal_reanalysis_max_air_temp_C from 28-32 weeks ago - the hotter, the lower cases.
- FE_station_diur_temp_rng_c from 23, 26-32 weeks ago - the bigger the gap between maximum & minimum temperatures, the more cases.
- FE_ndvi_nw of 28, 30-32 weeks ago - the higher the vegetation at north-west direction, the more cases.
- FE_station_min_temp_c of 6 weeks ago - the higher the minimum ground temperature, the more cases
- FE_LN_precipitation_amt_mm of 32 weeks ago - the more rainfall, the lesser the cases.

```
# sj_train, Model 2
# Lasso_sj_M2, X_sj_subtrain_M1 - Function call to Lasso_Coef(model, X_subtrain, X_label)
Lasso_Coef_sj_M2 = Lasso_Coef(Lasso_sj_M2, X_sj_subtrain_M2, X_label_sj_train_M2)
```

```
Non-zero coefficient in descending absolute value order:
                                        Coef
FE_normal_reanalysis_max_air_temp_C(t-29) -0.187649
FE_normal_reanalysis_max_air_temp_C(t-30) -0.130869
FE_station_diur_temp_rng_c(t-29)           0.125769
FE_station_diur_temp_rng_c(t-30)           0.120828
FE_normal_reanalysis_max_air_temp_C(t-31) -0.098969
FE_station_diur_temp_rng_c(t-28)           0.096412
FE_station_diur_temp_rng_c(t-31)           0.091030
FE_normal_reanalysis_max_air_temp_C(t-28) -0.090543
FE_normal_reanalysis_max_air_temp_C(t-32) -0.087704
FE_station_diur_temp_rng_c(t-32)           0.040954
FE_ndvi_nw(t-31)                           0.034419
FE_station_diur_temp_rng_c(t-26)           0.033460
FE_station_diur_temp_rng_c(t-27)           0.030560
FE_ndvi_nw(t-30)                           0.026436
FE_station_min_temp_c(t-6)                 0.017392
FE_ndvi_nw(t-32)                           0.012119
FE_station_diur_temp_rng_c(t-23)           0.011349
FE_LN_precipitation_amt_mm(t-32)          -0.010362
FE_ndvi_nw(t-28)                           0.007760
```

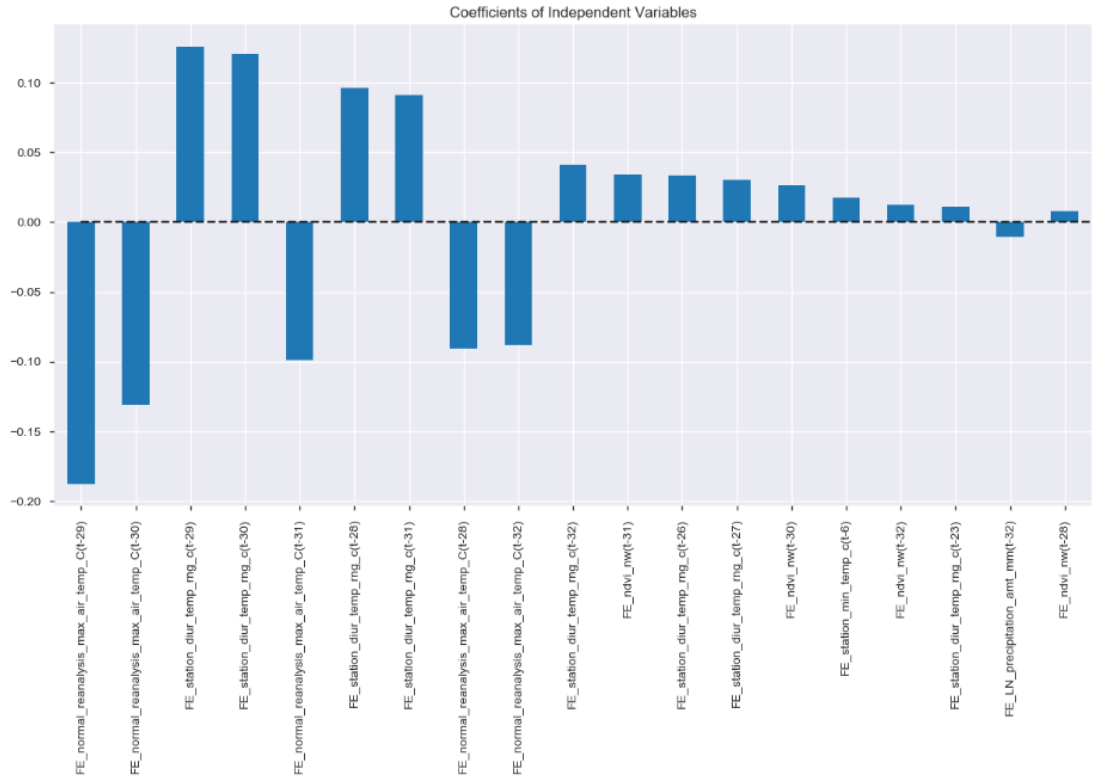


Figure 5.3H: M2 model – Lasso coefficients

M2 model's predictions for subtrain and subvalid datasets were matched with the corresponding row indices in Figure 5.3 I-J.

```
# PART 5F - Create Dataframe for Lasso Predictions

# sj_train, Model 2 - Function call to LassoPredict(Lasso_model, X, row_index, DependentVar)
Predict_sj_subtrain_M2 = LassoPredict(Lasso_sj_M2, X_sj_subtrain_M2, sj_index_subtrain_M2, DependentVar_M2)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 630 entries, sj1990-12-10 to sj2003-01-15
Data columns (total 1 columns):
Advance4W_normal_total_cases    630 non-null float64
dtypes: float64(1)
memory usage: 9.8+ KB
None
                     Advance4W_normal_total_cases
City_week_start_date
sj1990-12-10                             0.527609
sj1990-12-17                             0.495184
sj1990-12-24                             0.457250
sj1991-01-01                             0.483936
sj1991-01-08                             0.416035
                     Advance4W_normal_total_cases
City_week_start_date
sj2002-12-17                             0.064584
sj2002-12-24                             0.004769
sj2003-01-01                             0.000000
sj2003-01-08                             0.000000
sj2003-01-15                             0.000000
```

Figure 5.3I: M2 model – predictions for subtrain

```
# sj_train, Model 2 - Function call to LassoPredict(Lasso_model, X, row_index, DependentVar)
Predict_sj_subvalid_M2 = LassoPredict(Lasso_sj_M2, X_sj_subvalid_M2, sj_index_subvalid_M2, DependentVar_M2)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 270 entries, sj2003-01-22 to sj2008-03-25
Data columns (total 1 columns):
Advance4W_normal_total_cases    270 non-null float64
dtypes: float64(1)
memory usage: 4.2+ KB
None
                     Advance4W_normal_total_cases
City_week_start_date
sj2003-01-22                                  0.0
sj2003-01-29                                  0.0
sj2003-02-05                                  0.0
sj2003-02-12                                  0.0
sj2003-02-19                                  0.0
                     Advance4W_normal_total_cases
City_week_start_date
sj2008-02-26                                  0.0
sj2008-03-04                                  0.0
sj2008-03-11                                  0.0
sj2008-03-18                                  0.0
sj2008-03-25                                  0.0
```

Figure 5.3J: M2 model – predictions for subvalid

X and y were prepared from sj_test (Figure 5.3K) before predicting using M2 model (Figure 5.3L). M2 predictions for subtrain, subvalid and test datasets were merged in Figure 5.3M.

```
# Part 5A - Prepare Independent & Dependent Variables

# sj_train, Model M2 - Function call to Prepare_Xy(city, df, AddFE, DependentVar)
X_sj_test_M2, y_sj_test_M2, X_label_sj_test_M2 = Prepare_Xy(city_M2, sj_test, AddFE_M2, DependentVar_M2)
```

```
Number of rows in independent & dependent variables matched after removing NaN from 32 weeks lagged variables.


<class 'pandas.core.frame.DataFrame'>
Index: 264 entries, sj2008-04-01 to sj2013-04-23
Data columns (total 365 columns):
weekofyear                                264 non-null float64
FE_ndvi_nw(t)                             264 non-null float64
FE_ndvi_nw(t-1)                           264 non-null float64
FE_ndvi_nw(t-2)                           264 non-null float64
FE_ndvi_nw(t-3)                           264 non-null float64
FE_ndvi_nw(t-4)                           264 non-null float64
FE_ndvi_nw(t-5)                           264 non-null float64
FE_ndvi_nw(t-6)                           264 non-null float64
FE_ndvi_nw(t-7)                           264 non-null float64
FE_ndvi_nw(t-8)                           264 non-null float64
FE_ndvi_nw(t-9)                           264 non-null float64
FE_ndvi_nw(t-10)                          264 non-null float64
```

Figure 5.3K: M2 model – prepare X and y for sj_test

```
# PART 5F - Create Dataframe for Lasso Predictions

# Model 2, X_sj_test - Function call to LassoPredict(Lasso_model, X, row_index)
Predict_sj_test_M2 = LassoPredict(Lasso_sj_M2, X_sj_test_M2, sj_index_test, DependentVar_M2)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 264 entries, sj2008-04-01 to sj2013-04-23
Data columns (total 1 columns):
Advance4W_normal_total_cases    264 non-null float64
dtypes: float64(1)
memory usage: 4.1+ KB
None
                    Advance4W_normal_total_cases
City_week_start_date
sj2008-04-01                                 0.0
sj2008-04-08                                 0.0
sj2008-04-15                                 0.0
sj2008-04-22                                 0.0
sj2008-04-29                                 0.0
                    Advance4W_normal_total_cases
City_week_start_date
sj2013-03-26                                 0.0
sj2013-04-02                                 0.0
sj2013-04-09                                 0.0
sj2013-04-16                                 0.0
sj2013-04-23                                 0.0
```

Figure 5.3L: M2 model – predictions for sj_test

```
# PART 7A - Concatenate Lasso Predictions From y_sj_subtrain_M1, y_sj_subvalid_M1 & y_sj_test_M1

# sj, Model 2 - Function call to Merge_LassoPred (Pred_subtrain, Pred _subvalid, Pred _test, Model_number)
Lasso_prediction_M2 = Merge_LassoPred(Predict_sj_subtrain_M2, Predict_sj_subvalid_M2, Predict_sj_test_M2, 'M2')
```

```
                     Advance4W_normal_Lasso_pred_M2
City_week_start_date
sj1990-12-10                               0.527609
sj1990-12-17                               0.495184
sj1990-12-24                               0.457250
sj1991-01-01                               0.483936
sj1991-01-08                               0.416035


                     Advance4W_normal_Lasso_pred_M2
City_week_start_date
sj2013-03-26                                    0.0
sj2013-04-02                                    0.0
sj2013-04-09                                    0.0
sj2013-04-16                                    0.0
sj2013-04-23                                    0.0
```

Figure 5.3M: M2 model – merge predictions for subtrain, subvalid & test datasets

M2 predictions were merged with actual total_cases (Figure 5.3N) before plotting them in the same chart (Figure 5.3O). M2 predictions shows annual seasonality with peaks in yearend but failed to replicate the outbreaks in train dataset. The predictions using lagged features are smoother than base model, M1.

```
# sj, Model 2 - Function call to Merge_FEPredict_CityFE(df_FEPredict, df_cityFE)
sj_complete_M2 = Merge_FEPredict_CityFE(Lasso_prediction_M2, sj_completeFE)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1228 entries, sj1990-04-30 to sj2013-04-23
Columns: 390 entries, weekofyear to Advance4W_normal_Lasso_pred_M2
dtypes: float64(386), int32(1), object(3)
memory usage: 3.7+ MB


                     weekofyear week_start_date   ndvi_nw   ndvi_se  \
City_week_start_date
sj1990-04-30           -0.568946      1990-04-30  0.396319  0.368874
sj1990-05-07           -0.502406      1990-05-07  0.820552 -0.269711
sj1990-05-14           -0.435866      1990-05-14  1.160288 -0.360870
sj1990-05-21           -0.369326      1990-05-21  1.955795  0.882796
sj1990-05-28           -0.302786      1990-05-28  2.144833  1.300720

                     precipitation_amt_mm  reanalysis_precip_amt_kg_per_m2  \
City_week_start_date
sj1990-04-30                        12.42                            32.00
sj1990-05-07                        22.82                            17.94
```

Figure 5.3N: M2 model – merge predicted & actual total_cases

```
# sj, Model 2 - Function call to Plot_Actual_Predict(df_Pred_Features, city, Model_number)
Plot_Actual_Predict(sj_complete_M2, 'San Juan', 'M2')
```

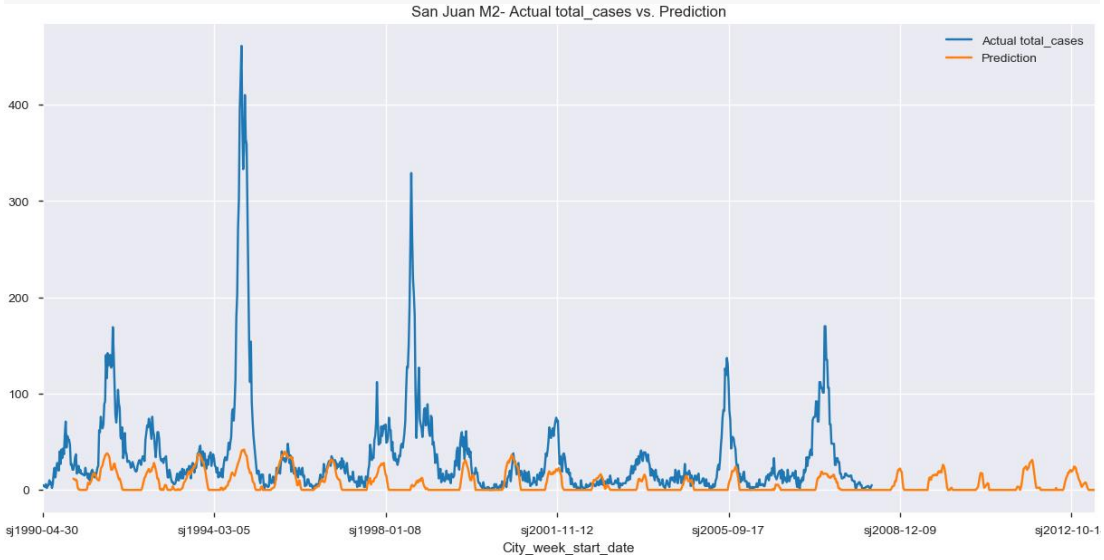


Figure 5.3O: M2 model – predicted vs. actual total_cases

5.4 Train Iquitos' M3 - Base Lasso Model without Feature Engineering

Please refer to the following sections in attached Jupyter notebook:

- PART 9 - Train Iquitos' Base Lasso Model
- PART 10 - Fit Train Iquitos' Base Lasso Model Using Testing Dataset

iq_train underwent X and y preparation (Figure 5.4A) prior to subtrain and subvalid split. According to Figure 5.4B, the subtrain ranges from 12-Feb-2001 to 6-Aug-2007 while the subvalid ranges from 13-Aug-2007 to 28-May-2010.

```
# Part 5A - Prepare Independent & Dependent Variables

# iq_train, Model M3 specification - most basic with no FE or ln for dependent variable
city_M3 = 'iq'
df_M3 = iq_train
AddFE_M3 = False
DependentVar_M3 = 'Advance4W_normal_total_cases'

# iq_train, Model 3 - Function call to Prepare_Xy(city, df, AddFE, DependentVar)
X_iq_train_M3, y_iq_train_M3, X_label_iq_train_M3 = Prepare_Xy(city_M3, df_M3, AddFE_M3, DependentVar_M3)
```

```
Number of rows in independent & dependent variables matched after removing NaN from 32 weeks lagged variables.


<class 'pandas.core.frame.DataFrame'>
Index: 484 entries, iq2001-02-12 to iq2010-05-28
Data columns (total 16 columns):
weekofyear                              484 non-null float64
ndvi_ne                                 484 non-null float64
ndvi_sw                                 484 non-null float64
precipitation_amt_mm                    484 non-null float64
LN_reanalysis_precip_amt_kg_per_m2      484 non-null float64
reanalysis_relative_humidity_percent    484 non-null float64
reanalysis_specific_humidity_g_per_kg   484 non-null float64
station_avg_temp_c                      484 non-null float64
station_diur_temp_rng_c                 484 non-null float64
station_max_temp_c                      484 non-null float64
station_min_temp_c                      484 non-null float64
LN_station_precip_mm                    484 non-null float64
normal_reanalysis_max_air_temp_C        484 non-null float64
normal_reanalysis_min_air_temp_C        484 non-null float64
normal_reanalysis_avg_temp_C            484 non-null float64
normal_reanalysis_tdtr_C                484 non-null float64
dtypes: float64(16)
memory usage: 64.3+ KB
```

Figure 5.4A: M3 – prepare X and y

```
# Part 5B - Prepare Training & Validation Datasets at Simple 70:30 split

# iq_train, Model 3
# X_iq_train_M3, y_iq_train_M3, X_label_iq_train_M3 - Function call to TrainValidSplit(X, y)
X_iq_subtrain_M3, y_iq_subtrain_M3, X_iq_subvalid_M3, y_iq_subvalid_M3, iq_index_subtrain_M3, iq_index_subvalid_M3 = TrainValidS
```

```
SubTrain period - row indices:
Index(['iq2001-02-12', 'iq2001-02-19', 'iq2001-02-26', 'iq2001-03-05',
       'iq2001-03-12', 'iq2001-03-19', 'iq2001-03-26', 'iq2001-04-02',
       'iq2001-04-09', 'iq2001-04-16',
       ...
       'iq2007-06-04', 'iq2007-06-11', 'iq2007-06-18', 'iq2007-06-25',
       'iq2007-07-02', 'iq2007-07-09', 'iq2007-07-16', 'iq2007-07-23',
       'iq2007-07-30', 'iq2007-08-06'],
      dtype='object', name='City_week_start_date', length=338)

SubValid period - row indices:
Index(['iq2007-08-13', 'iq2007-08-20', 'iq2007-08-27', 'iq2007-09-03',
       'iq2007-09-10', 'iq2007-09-17', 'iq2007-09-24', 'iq2007-10-01',
       'iq2007-10-08', 'iq2007-10-15',
       ...
       'iq2010-03-26', 'iq2010-04-02', 'iq2010-04-09', 'iq2010-04-16',
       'iq2010-04-23', 'iq2010-04-30', 'iq2010-05-07', 'iq2010-05-14',
       'iq2010-05-21', 'iq2010-05-28'],
      dtype='object', name='City_week_start_date', length=146)
```

Figure 5.4B: M3 – subtrain & subvalid data split

The fitting of BasicLasso1 using alpha of 0.1 in Figure 5.4C indicated that 4/15 of the predictors have non-zero coefficients.

```
# Part 5C - Basic Lasso Using Data Split (70:30) & Alpha = 0.1

# iq_subtrain, Model 3 - Function call for Basic_Lasso1(X_subtrain, y_subtrain, X_subvalid, y_subvalid, X_label)
Basic_Lasso1(X_iq_subtrain_M3, y_iq_subtrain_M3, X_iq_subvalid_M3, y_iq_subvalid_M3, X_label_iq_train_M3)
```

```
Alpha:  0.1
Subtrain MAE =  1.7303
Subvalid MAE =  2.4149
Time lapsed:  0.002 seconds
```

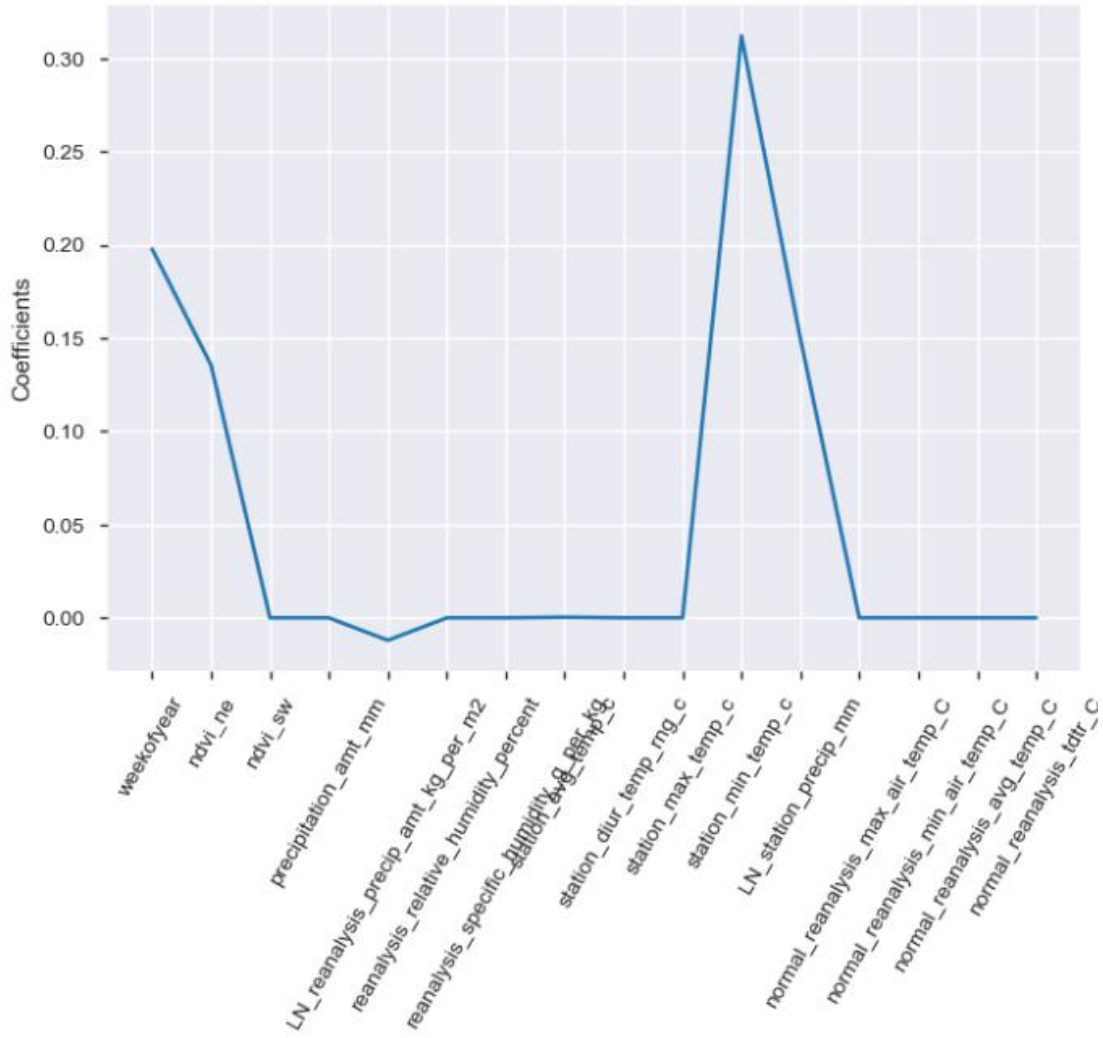


Figure 5.4C: M3 – BasicLasso1

Lasso coefficient path in Figure 5.4D showed that

- Initial coefficients ranges from (-0.6 to +0.6).
- The magnitude of coefficients decrease as alpha increases.
- All coefficients zerrolized as alpha approaches 1

```
# Part 5D - Basic Lasso (Simple 70:30 Data Split) Over Different Alphas
# Set alpha range
Alphas2_M3 = np.logspace(-2, 1, 50)

# iq_train, Model 3 - Function call to Basic_Lasso2(Alpha2, X_subtrain, y_subtrain)
Basic_Lasso2(Alphas2_M3, X_iq_subtrain_M3, y_iq_subtrain_M3)
```

```
Time lapsed:  0.057 seconds
```

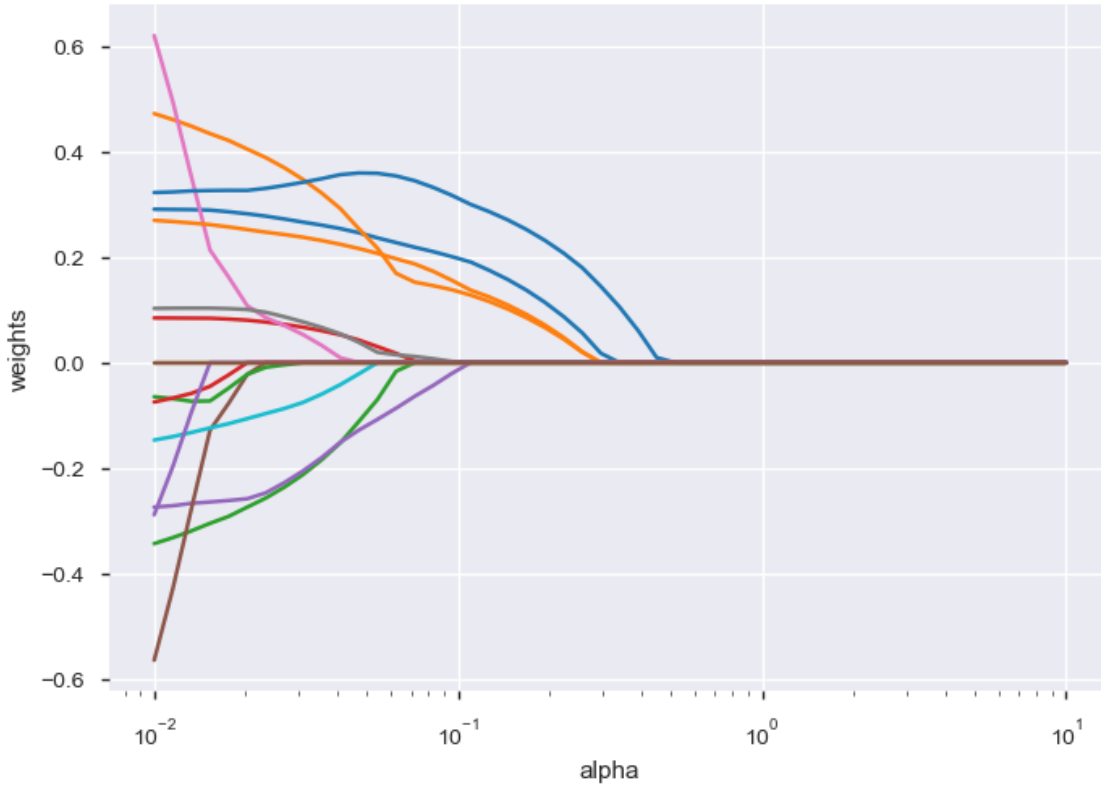


Figure 5.4D: M3 – Lasso coefficient path

The initial grid search Figure 5.4E: M3 recorded no optimal solution as the negative MAE chart peaked and flatten for alpha => 0.7564633275546291. This corresponds to weight = 0 in Figure 5.4D. This will cause model with insignificant coefficients and zero predictions which are not useful at all.

```
# Part 5E - Optimize Lasso Using GridSearchCV with 10-splits from TimeSeriesSplit
Alphas3_M3_init = np.logspace(-2, 1, 100)

# iq_train, Model 3 - Function call to Lasso_GridSearchCV_TimeSeriesSplit(Alphas3, X_subtrain, y_subtrain, X_subvalid, y_subvali
Lasso_iq_M3_init = Lasso_GridSearchCV_TimeSeriesSplit(Alphas3_M3_init, X_iq_subtrain_M3, y_iq_subtrain_M3, X_iq_subvalid_M3, y_i
```

```
Optimized alpha:  0.7564633275546291
Subtrain MAE =  1.7217
Subvalid MAE =  2.3488
Time lapsed:  1.8573  seconds
```

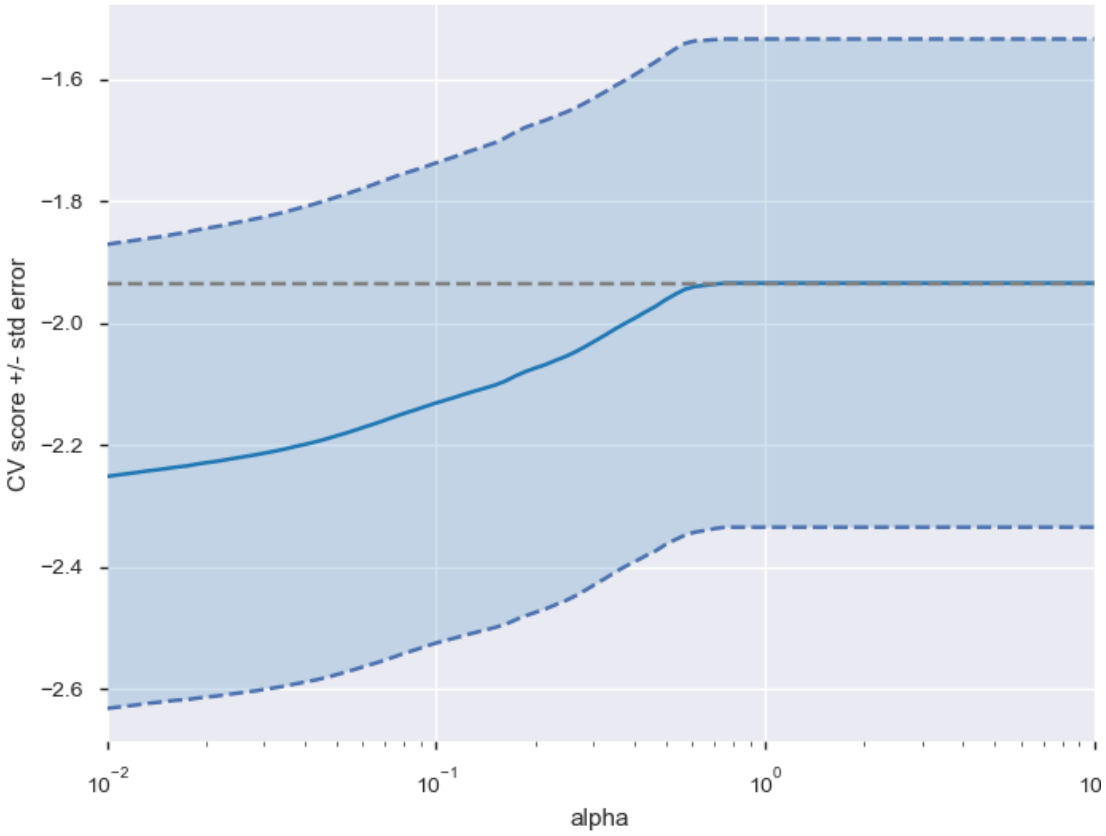


Figure 5.4E: M3 – initial grid search

Therefore, we need to fine tune the alphas to be the range of 0.01-0.1 where there are at least a handful of non-zero weights. According to Figure 5.4F, an unique solution was found at alpha = 0.1 while MAE deteriorated from subtrain 1.7303 to subvalid 2.4149.

```
# Choose alpha range where there are at least a handful of predictors will remain
Alphas3_M3 = np.logspace(-2, -1, 100)

# iq_train, Model 3 - Function call to Lasso_GridSearchCV_TimeSeriesSplit(Alphas3, X_subtrain, y_subtrain, X_subvalid, y_subvali
Lasso_iq_M3 = Lasso_GridSearchCV_TimeSeriesSplit(Alphas3_M3, X_iq_subtrain_M3, y_iq_subtrain_M3, X_iq_subvalid_M3, y_iq_subvalid_
```

```
Optimized alpha:  0.1
Subtrain MAE =  1.7303
Subvalid MAE =  2.4149
Time lapsed:  2.0424  seconds
```

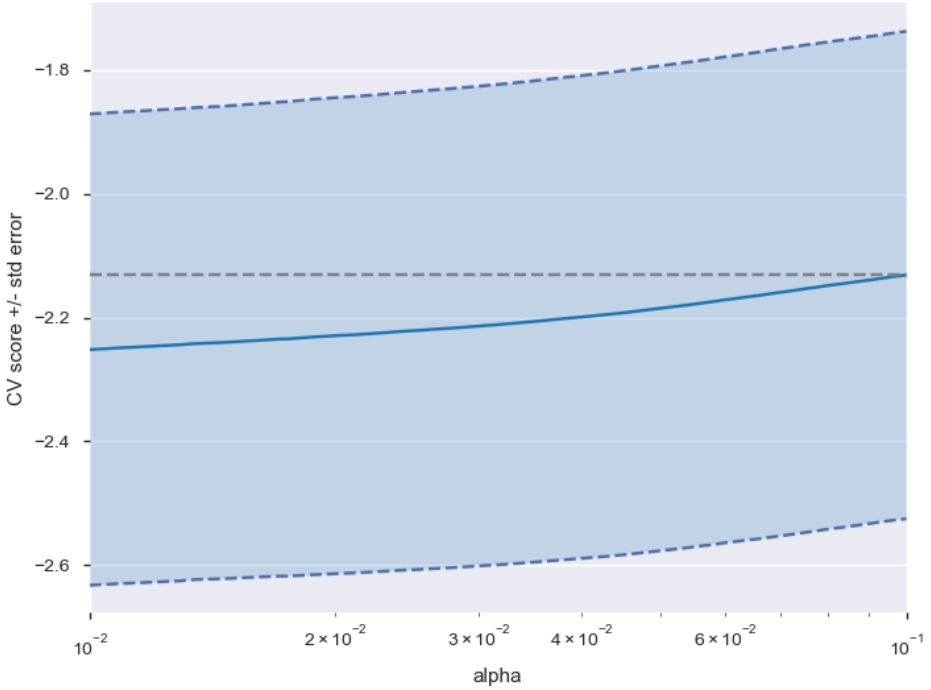


Figure 5.4F: M3 – refined grid search

According to Figure 5.4G, 6/15 predictors were found to be significant in predicting Advance4W_normal_total_cases with non-zero coefficients. They are

- station_min_temp_c - the hotter, the more cases.
- weekofyear - cases increase toward year end.
- LN_station_precip_mm - the more rainfall, the more cases.
- ndvi_ne - the more vegetation on the north-east direction, the more cases.
- LN_reanalysis_precip_amt_kg_per_m2 - the less rainfall, the more cases
- station_avg_temp_c - the hotter, the more cases.

```
# iq_train, Model 3
# Lasso_iq_M3, X_iq_subtrain_M3 - Function call to Lasso_Coef(model, X_subtrain, X_label)
Lasso_Coef_iq_M3 = Lasso_Coef(Lasso_iq_M3, X_iq_subtrain_M3, X_label_iq_train_M3)
```

```
Non-zero coefficient in descending absolute value order:
                                    Coef
station_min_temp_c                 0.312148
weekofyear                         0.197789
LN_station_precip_mm               0.149914
ndvi_ne                            0.135542
LN_reanalysis_precip_amt_kg_per_m2 -0.012013
station_avg_temp_c                 0.000366
```

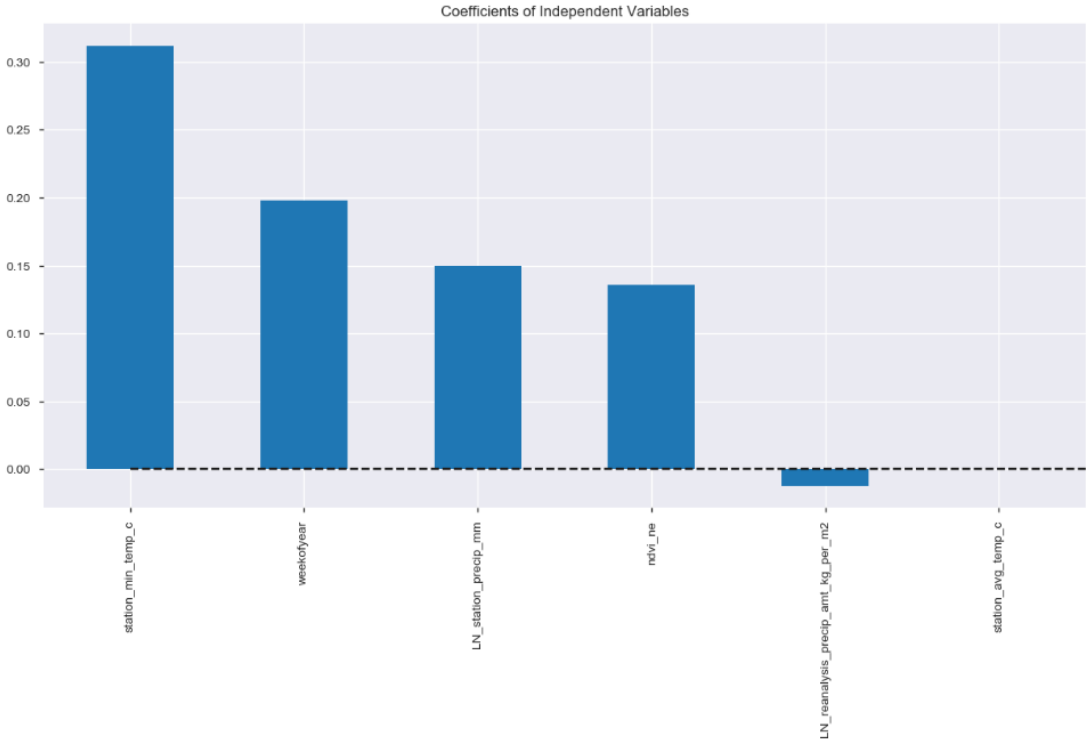


Figure 5.4G: M3 – Lasso coefficients

M3 model's predictions for subtrain and subvalid are found in Figure 5.4 H-I.

```
# PART 5F - Create Dataframe for Lasso Predictions
# iq_train, Model 3 - Function call to LassoPredict(Lasso_model, X, row_index, DependentVar)
Predict_iq_subtrain_M3 = LassoPredict(Lasso_iq_M3, X_iq_subtrain_M3, iq_index_subtrain_M3, DependentVar_M3)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 338 entries, iq2001-02-12 to iq2007-08-06
Data columns (total 1 columns):
Advance4W_normal_total_cases    338 non-null float64
dtypes: float64(1)
memory usage: 5.3+ KB
None
                     Advance4W_normal_total_cases
City_week_start_date
iq2001-02-12                                  0.0
iq2001-02-19                                  0.0
iq2001-02-26                                  0.0
iq2001-03-05                                  0.0
iq2001-03-12                                  0.0
                     Advance4W_normal_total_cases
City_week_start_date
iq2007-07-09                                  0.0
iq2007-07-16                                  0.0
iq2007-07-23                                  0.0
iq2007-07-30                                  0.0
iq2007-08-06                                  0.0
```

Figure 5.4H: M3 – predictions for subtrain

```
# iq_train, Model 3 - Function call to LassoPredict(Lasso_model, X, row_index, DependentVar)
Predict_iq_subvalid_M3 = LassoPredict(Lasso_iq_M3, X_iq_subvalid_M3, iq_index_subvalid_M3, DependentVar_M3)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 146 entries, iq2007-08-13 to iq2010-05-28
Data columns (total 1 columns):
Advance4W_normal_total_cases    146 non-null float64
dtypes: float64(1)
memory usage: 2.3+ KB
None
                     Advance4W_normal_total_cases
City_week_start_date
iq2007-08-13                             0.220424
iq2007-08-20                             0.000000
iq2007-08-27                             0.000000
iq2007-09-03                             0.079317
iq2007-09-10                             0.399240
                     Advance4W_normal_total_cases
City_week_start_date
iq2010-04-30                             0.000000
iq2010-05-07                             0.000000
iq2010-05-14                             0.000000
iq2010-05-21                             0.000000
iq2010-05-28                             0.330528
```

Figure 5.4I: M3 – predictions for subvalid

The test dataset of Iquitos was preprocessed via

- Figure 5.4J - rescaling
- Figure 5.4K - single imputation of MICE
- Figure 5.4L - removal of highly correlated variables
- Figure 5.4M - log transformation of right-skewed variables
- Figure 5.4R - standardization of numerical variables

```
# PART 6A - Preprocess iq_test
# PART 2F - Rescaling of variables (temperatures)
# iq_test - Function call to rescaling(df)
rescaling(iq_test)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 160 entries, iq2010-06-04 to iq2013-06-25
Data columns (total 29 columns):
year                                    160 non-null int64
weekofyear                              160 non-null int64
week_start_date                         160 non-null object
ndvi_ne                                 160 non-null float64
ndvi_nw                                 160 non-null float64
ndvi_se                                 160 non-null float64
ndvi_sw                                 160 non-null float64
precipitation_amt_mm                    160 non-null float64
reanalysis_precip_amt_kg_per_m2         160 non-null float64
reanalysis_relative_humidity_percent    160 non-null float64
reanalysis_sat_precip_amt_mm            160 non-null float64
reanalysis_specific_humidity_g_per_kg   160 non-null float64
station_avg_temp_c                      150 non-null float64
station_diur_temp_rng_c                 150 non-null float64
station_max_temp_c                      159 non-null float64
station_min_temp_c                      153 non-null float64
```

Figure 5.4J: M3 – rescaling

```
# PART 3C - Missing Values Analysis & Treatment
# iq_test - Function call to MICE_imputation(df, list_variables)
MICE_imputation(iq_test, list_test_variables)

# Important that                      - Advance4W_normal_total_cases            0 non-null float64
# Correct since these ar the unlaggeds - total_cases                            4 non-null float64
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 160 entries, iq2010-06-04 to iq2013-06-25
Data columns (total 29 columns):
year                                    160 non-null float64
weekofyear                              160 non-null float64
week_start_date                         160 non-null object
ndvi_ne                                 160 non-null float64
ndvi_nw                                 160 non-null float64
ndvi_se                                 160 non-null float64
ndvi_sw                                 160 non-null float64
precipitation_amt_mm                    160 non-null float64
reanalysis_precip_amt_kg_per_m2         160 non-null float64
reanalysis_relative_humidity_percent    160 non-null float64
reanalysis_sat_precip_amt_mm            160 non-null float64
reanalysis_specific_humidity_g_per_kg   160 non-null float64
station_avg_temp_c                      160 non-null float64
station_diur_temp_rng_c                 160 non-null float64
station_max_temp_c                      160 non-null float64
station_min_temp_c                      160 non-null float64
station_precip_mm                       160 non-null float64
total_cases                             4 non-null float64
dataset_label                           160 non-null object
Population_inter                        160 non-null int32
```

Figure 5.4K: M3 – single imputation of MICE

```
# PART 4A - Feature Selection (Round 1 - Based on EDA)
# Drop the same variables as per iq_train
iq_test.drop(list_iq_train_drop, axis=1, inplace = True)

iq_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 160 entries, iq2010-06-04 to iq2013-06-25
Data columns (total 22 columns):
weekofyear                             160 non-null float64
week_start_date                        160 non-null object
ndvi_ne                                160 non-null float64
ndvi_sw                                160 non-null float64
precipitation_amt_mm                   160 non-null float64
reanalysis_precip_amt_kg_per_m2        160 non-null float64
reanalysis_relative_humidity_percent   160 non-null float64
reanalysis_specific_humidity_g_per_kg  160 non-null float64
station_avg_temp_c                     160 non-null float64
station_diur_temp_rng_c                160 non-null float64
station_max_temp_c                     160 non-null float64
station_min_temp_c                     160 non-null float64
station_precip_mm                      160 non-null float64
total_cases                            4 non-null float64
dataset_label                          160 non-null object
Population_inter                       160 non-null int32
city                                   160 non-null object
Advance4W_normal_total_cases           0 non-null float64
normal_reanalysis_max_air_temp_C       160 non-null float64
normal_reanalysis_min_air_temp_C       160 non-null float64
normal_reanalysis_avg_temp_C           160 non-null float64
normal_reanalysis_tdtr_C               160 non-null float64
dtypes: float64(18), int32(1), object(3)
```

Figure 5.4L: M3 – removal of highly correlated variables

```
# PART 4B - Logarithmic Transformation
# iq_test - Function call to LogTransformation_iq(df, TestBoolean)
LogTransformation_iq(iq_test, TestBoolean = True)
```

```
station_avg_temp_c                     160 non-null float64
station_diur_temp_rng_c                160 non-null float64
station_max_temp_c                     160 non-null float64
station_min_temp_c                     160 non-null float64
station_precip_mm                      160 non-null float64
total_cases                            4 non-null float64
dataset_label                          160 non-null object
Population_inter                       160 non-null int32
city                                   160 non-null object
Advance4W_normal_total_cases           0 non-null float64
normal_reanalysis_max_air_temp_C       160 non-null float64
normal_reanalysis_min_air_temp_C       160 non-null float64
normal_reanalysis_avg_temp_C           160 non-null float64
normal_reanalysis_tdtr_C               160 non-null float64
LN_reanalysis_precip_amt_kg_per_m2     160 non-null float64
LN_station_precip_mm                   159 non-null float64
LN_Advance4W_normal_total_cases        0 non-null float64
dtypes: float64(21), int32(1), object(3)
memory usage: 31.9+ KB
```

Figure 5.4M: M3 – log transformation of right-skewed variables

LN_station_precip_mm has one NaN (Figure 5.4M) which was traced to 27-Aug-2010 (Figure 5.4N) when the single MICE imputatation of station_precip_mm has a negative value of -2.0891(Figure 5.4O). It was rectified by overwriting negative station_precip_mm and NaN LN_station_precip_mm is NaN with zeros (Figure 5.4O).

```
# Check which row of iq_test has 'LN_station_precip_mm' = NaN
print(iq_test[iq_test['LN_station_precip_mm'].isnull()])
```

```
                     weekofyear week_start_date   ndvi_ne   ndvi_sw  \
City_week_start_date
iq2010-08-27               34.0      2010-08-27  0.319043  0.344057

                     precipitation_amt_mm  reanalysis_precip_amt_kg_per_m2  \
City_week_start_date
iq2010-08-27                         7.82                            14.21

                     reanalysis_relative_humidity_percent  \
City_week_start_date
iq2010-08-27                                    68.615714

                     reanalysis_specific_humidity_g_per_kg  \
City_week_start_date
iq2010-08-27                                     14.484286

                     station_avg_temp_c  station_diur_temp_rng_c  ...  \
City_week_start_date                                              ...
iq2010-08-27                       27.4                     13.1  ...

                     Population_inter  city  Advance4W_normal_total_cases  \
City_week_start_date
iq2010-08-27                   450930    iq                           NaN
```

Figure 5.4N: M3 – check when LN_station_precip_mm has NaN

```
# Check the corresponding 'station_precip_mm' for the row with 'LN_station_precip_mm' = NaN
print(iq_test.loc['iq2010-08-27', ['station_precip_mm','LN_station_precip_mm']])
```

```
station_precip_mm      -2.0891
LN_station_precip_mm       NaN
Name: iq2010-08-27, dtype: object
```

Figure 5.4O: M3 – check the value of station_precip_mm when LN_station_precip_mm is NaN

```
# Manually overwrite the 'station_precip_mm' & 'LN_station_precip_mm' in both iq_test & X_iq_test_M3
iq_test.loc['iq2010-08-27', 'station_precip_mm'] = 0
iq_test.loc['iq2010-08-27', 'LN_station_precip_mm'] = 0

# Check the replaced values
print(iq_test.loc['iq2010-08-27', ['station_precip_mm','LN_station_precip_mm']])
```

```
station_precip_mm       0
LN_station_precip_mm    0
Name: iq2010-08-27, dtype: object
```

Figure 5.4P: M3 – replace negative station_precip_mm and NaN LN_station_precip_mm is NaN with zeros

```
# PART 4C - Standardization Transformation

# iq_test - Function call to Standardization(df, list_df) using the same list_standardize_sj
iq_test_Standard = Standardization(iq_test, list_shortlist_iq)

# Descriptive statistics, round to 2 decimal places
# The output should have zero means & 1 standard deviations
round(iq_test_Standard.describe(), 2)

iq_test = iq_test_Standard
```

Figure 5.4Q: M3 – standardization

The bottom 32 rows from iq_train to be on top of iq_test in order to create 32 weeks lags (Figure 5.4 R-S).

```
# PART 6B - Feature Engineering for iq_test
# iq - Function call to Lengthen_Test(df_train, df_test, city)
iq_test = Lengthen_Test(iq_train, iq_test, 'iq')
```

```
Lengthened iq_test:


<class 'pandas.core.frame.DataFrame'>
Index: 192 entries, iq2009-10-22 to iq2013-06-25
Data columns (total 25 columns):
weekofyear                              192 non-null float64
week_start_date                         192 non-null object
ndvi_ne                                 192 non-null float64
ndvi_sw                                 192 non-null float64
precipitation_amt_mm                    192 non-null float64
reanalysis_precip_amt_kg_per_m2         192 non-null float64
reanalysis_relative_humidity_percent    192 non-null float64
reanalysis_specific_humidity_g_per_kg   192 non-null float64
station_avg_temp_c                      192 non-null float64
station_diur_temp_rng_c                 192 non-null float64
station_max_temp_c                      192 non-null float64
station_min_temp_c                      192 non-null float64
```

Figure 5.4R: Iquitos test dataset – add bottom 32 weeks from iq_train

```
# PART 4E - Feature Engineering Based on CCF Analysis in R
# sj_test - Function call for FeatureEng(df, dict_df)
sj_test, list_FE_sj_train = FeatureEng(sj_test, dict_sj_train)
```

```
Top 33 rows of new columns
```

City_week_start_date	FE_ndvi_nw(t)	FE_ndvi_nw(t-1)	FE_ndvi_nw(t-2)	\
sj2007-08-20	0.274676	NaN	NaN	
sj2007-08-27	-0.044557	0.274676	NaN	
sj2007-09-03	-0.270373	-0.044557	0.274676	
sj2007-09-10	-0.120319	-0.270373	-0.044557	
sj2007-09-17	0.078281	-0.120319	-0.270373	
sj2007-09-24	-0.780556	0.078281	-0.120319	
sj2007-10-01	-0.474858	-0.780556	0.078281	
sj2007-10-08	0.399353	-0.474858	-0.780556	
sj2007-10-15	0.715643	0.399353	-0.474858	
sj2007-10-22	-0.849074	0.715643	0.399353	
sj2007-10-29	-0.149006	-0.849074	0.715643	
sj2007-11-05	-1.284340	-0.149006	-0.849074	
sj2007-11-12	-1.438256	-1.284340	-0.149006	
sj2007-11-19	-0.899542	-1.438256	-1.284340	

Figure 5.4S: Iquitos test dataset – feature engineering

Iquitos test dataset underwent X and y preparation (Figure 5.4T) before generating M3 predictions (Figure 5.4U).

```
# Part 5A - Prepare Independent & Dependent Variables
# iq_train, Model M3 - Function call to Prepare_Xy(city, df, AddFE, DependentVar)
X_iq_test_M3, y_iq_test_M3, X_label_iq_test_M3 = Prepare_Xy(city_M3, iq_test, AddFE_M3, DependentVar_M3)
```

```
Number of rows in independent & dependent variables matched after removing NaN from 32 weeks lagged variables.


<class 'pandas.core.frame.DataFrame'>
Index: 160 entries, iq2010-06-04 to iq2013-06-25
Data columns (total 16 columns):
weekofyear                              160 non-null float64
ndvi_ne                                 160 non-null float64
ndvi_sw                                 160 non-null float64
precipitation_amt_mm                    160 non-null float64
LN_reanalysis_precip_amt_kg_per_m2      160 non-null float64
reanalysis_relative_humidity_percent    160 non-null float64
reanalysis_specific_humidity_g_per_kg   160 non-null float64
station_avg_temp_c                      160 non-null float64
station_diur_temp_rng_c                 160 non-null float64
station_max_temp_c                      160 non-null float64
station_min_temp_c                      160 non-null float64
LN_station_precip_mm                    160 non-null float64
normal_reanalysis_max_air_temp_C        160 non-null float64
normal_reanalysis_min_air_temp_C        160 non-null float64
normal_reanalysis_avg_temp_C            160 non-null float64
normal_reanalysis_tdtr_C                160 non-null float64
dtypes: float64(16)
memory usage: 21.2+ KB
```

Figure 5.4T: Iquitos test dataset – prepare X and y

```
# PART 5F - Create Dataframe for Lasso Predictions
# Model 3, X_iq_test - Function call to LassoPredict(Lasso_model, X, row_index)
Predict_iq_test_M3 = LassoPredict(Lasso_iq_M3, X_iq_test_M3, iq_index_test, DependentVar_M3)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 160 entries, iq2010-06-04 to iq2013-06-25
Data columns (total 1 columns):
Advance4W_normal_total_cases    160 non-null float64
dtypes: float64(1)
memory usage: 2.5+ KB
None
                     Advance4W_normal_total_cases
City_week_start_date
iq2010-06-04                             0.000000
iq2010-06-11                             0.000000
iq2010-06-18                             0.035396
iq2010-06-25                             0.315532
iq2010-07-02                             0.000000
                     Advance4W_normal_total_cases
City_week_start_date
iq2013-05-28                             0.240965
iq2013-06-04                             0.165472
iq2013-06-11                             0.255366
iq2013-06-18                             0.339466
iq2013-06-25                             0.317868
```

Figure 5.4U: M3 – predictions for test dataset

M3 predictions for subtrain, subvalid and test datasets were merged (Figure 5.4V) before joining with actual total_cases (Figure 5.4W).

```
# PART 7A
# - Concatenate Lasso Predictions From y_iq_subtrain_M3, y_iq_subvalid_M3 & y_iq_test_M3

# iq, Model 3 - Function call to Merge_LassoPred (Pred_subtrain, Pred _subvalid, Pred _test, Model_number)
Lasso_prediction_M3 = Merge_LassoPred(Predict_iq_subtrain_M3, Predict_iq_subvalid_M3, Predict_iq_test_M3, 'M3')
```

```
                     Advance4W_normal_Lasso_pred_M3
City_week_start_date
iq2001-02-12                                    0.0
iq2001-02-19                                    0.0
iq2001-02-26                                    0.0
iq2001-03-05                                    0.0
iq2001-03-12                                    0.0


                     Advance4W_normal_Lasso_pred_M3
City_week_start_date
iq2013-05-28                               0.240965
iq2013-06-04                               0.165472
iq2013-06-11                               0.255366
iq2013-06-18                               0.339466
iq2013-06-25                               0.317868
```

Figure 5.4V: M3 – merger of subtrain, subvalid and test predictions

```
# PART 7B - Merge Lasso_prediction_M1 with sj_complete

# iq, Model 3 - Function call to Merge_BasePredict_CityBase(df_BasePredict, df_city)
iq_complete_M3 = Merge_BasePredict_CityBase(iq, Lasso_prediction_M3)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 676 entries, iq2000-07-01 to iq2013-06-25
Data columns (total 30 columns):
Advance4W_normal_Lasso_pred_M3          644 non-null float64
year                                    676 non-null int64
weekofyear                              676 non-null int64
week_start_date                         676 non-null object
ndvi_ne                                 673 non-null float64
ndvi_nw                                 673 non-null float64
ndvi_se                                 673 non-null float64
ndvi_sw                                 673 non-null float64
precipitation_amt_mm                    672 non-null float64
reanalysis_air_temp_k                   672 non-null float64
reanalysis_avg_temp_k                   672 non-null float64
reanalysis_dew_point_temp_k             672 non-null float64
reanalysis_max_air_temp_k               672 non-null float64
reanalysis_min_air_temp_k               672 non-null float64
reanalysis_precip_amt_kg_per_m2         672 non-null float64
reanalysis_relative_humidity_percent    672 non-null float64
```

Figure 5.4W: M3 – merger of predicted and actual total_case

M3 predicted vs. actual total_cases were plotted in Figure 5.4X whereby the predictions were much lower compared to the actual cases.

```
# iq, Model 3 - Function call to Plot_Actual_Predict(df_Pred_Features, city, Model_number)
Plot_Actual_Predict(iq_complete_M3, 'Iquitos', 'M3')
```

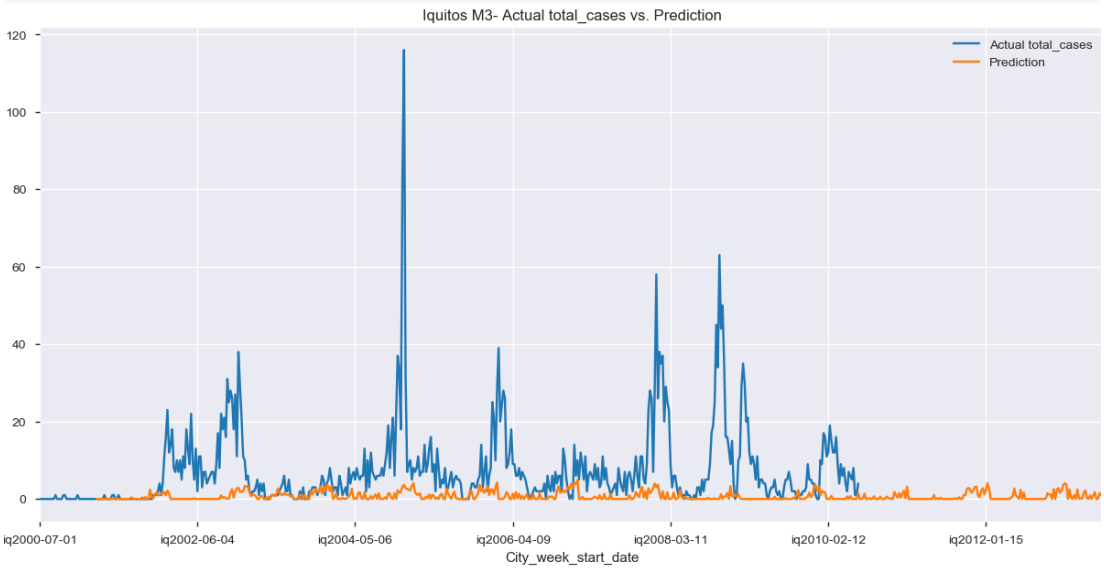


Figure 5.4X: M3 – predicted vs. actual total_case

5.5 Iquitos' M4 - Lasso Model with Feature Engineering from Lagged Climatic Variables

Please refer to “Part 11 - Iquitos' Featured Engineered Lasso Model” in attached Jupyter notebook.

iq_train and iq_test were merged (Figure 5.5A) to get a dataset with engineered features, before preparing X and y (Figure 5.5B), followed by the splitting into subtrain vs. subvalid datasets (Figure 5.5C).

```
# iq - Function call to Merge_TrainTestFE(df_train, df_test)
iq_completeFE = Merge_TrainTestFE(iq_train, iq_test)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 708 entries, iq2000-07-01 to iq2013-06-25
Data columns (total 266 columns):
weekofyear                                  708 non-null float64
week_start_date                             708 non-null object
ndvi_ne                                     708 non-null float64
ndvi_sw                                     708 non-null float64
precipitation_amt_mm                        708 non-null float64
reanalysis_precip_amt_kg_per_m2             708 non-null float64
reanalysis_relative_humidity_percent        708 non-null float64
reanalysis_specific_humidity_g_per_kg       708 non-null float64
station_avg_temp_c                          708 non-null float64
station_diur_temp_rng_c                     708 non-null float64
station_max_temp_c                          708 non-null float64
station_min_temp_c                          708 non-null float64
station_precip_mm                           708 non-null float64
total_cases                                 552 non-null float64
dataset_label                               708 non-null object
Population_inter                            708 non-null int32
```

Figure 5.5A: Merge iq_train and iq_test

```
# Part 5A - Prepare Independent & Dependent Variables

# Model M4 specification - most basic with no FE or ln for dependent variable
city_M4 = 'iq'
df_M4 = iq_train
AddFE_M4 = True
DependentVar_M4 = 'Advance4W_normal_total_cases'

# Model M1 - Function call to Prepare_Xy(city, df, AddFE, DependentVar)
X_iq_train_M4, y_iq_train_M4, X_label_iq_train_M4 = Prepare_Xy(city_M4, df_M4, AddFE_M4, DependentVar_M4)
```

```
Number of rows in independent & dependent variables matched after removing NaN from 32 weeks lagged variables.


<class 'pandas.core.frame.DataFrame'>
Index: 484 entries, iq2001-02-12 to iq2010-05-28
Data columns (total 242 columns):
weekofyear                                    484 non-null float64
FE_ndvi_ne(t-7)                               484 non-null float64
FE_ndvi_ne(t-8)                               484 non-null float64
FE_ndvi_ne(t-10)                              484 non-null float64
FE_ndvi_ne(t-11)                              484 non-null float64
FE_ndvi_ne(t-31)                              484 non-null float64
FE_ndvi_ne(t-32)                              484 non-null float64
FE_ndvi_sw(t-4)                               484 non-null float64
FE_ndvi_sw(t-5)                               484 non-null float64
FE_ndvi_sw(t-6)                               484 non-null float64
FE_ndvi_sw(t-7)                               484 non-null float64
FE_ndvi_sw(t-8)                               484 non-null float64
```

Figure 5.5B: M4 – prepare X and y

```
# Part 5B - Prepare Training & Validation Datasets at Simple 70:30 split

# X_iq_train_M4, y_iq_train_M4- Function call to TrainValidSplit(X, y)
X_iq_subtrain_M4, y_iq_subtrain_M4, X_iq_subvalid_M4, y_iq_subvalid_M4, iq_index_subtrain_M4, iq_index_subvalid_M4 = TrainValidSp
```

```
SubTrain period - row indices:
Index(['iq2001-02-12', 'iq2001-02-19', 'iq2001-02-26', 'iq2001-03-05',
       'iq2001-03-12', 'iq2001-03-19', 'iq2001-03-26', 'iq2001-04-02',
       'iq2001-04-09', 'iq2001-04-16',
       ...
       'iq2007-06-04', 'iq2007-06-11', 'iq2007-06-18', 'iq2007-06-25',
       'iq2007-07-02', 'iq2007-07-09', 'iq2007-07-16', 'iq2007-07-23',
       'iq2007-07-30', 'iq2007-08-06'],
      dtype='object', name='City_week_start_date', length=338)

SubValid period - row indices:
Index(['iq2007-08-13', 'iq2007-08-20', 'iq2007-08-27', 'iq2007-09-03',
       'iq2007-09-10', 'iq2007-09-17', 'iq2007-09-24', 'iq2007-10-01',
       'iq2007-10-08', 'iq2007-10-15',
       ...
       'iq2010-03-26', 'iq2010-04-02', 'iq2010-04-09', 'iq2010-04-16',
       'iq2010-04-23', 'iq2010-04-30', 'iq2010-05-07', 'iq2010-05-14',
       'iq2010-05-21', 'iq2010-05-28'],
      dtype='object', name='City_week_start_date', length=146)
```

Figure 5.5C: M4 – subtrain and subvalid data split

According to Figure 5.5D, many coefficients were zerolised in Basic_Lasso1 fitting using alpha of 0.1.

```
# Part 5C - Basic Lasso Using Data Split (70:30) & Alpha = 0.1

# iq, Model 4- Function call for Basic_Lasso1(X_subtrain, y_subtrain, X_subvalid, y_subvalid, X_label)
Basic_Lasso1(X_iq_subtrain_M4, y_iq_subtrain_M4, X_iq_subvalid_M4, y_iq_subvalid_M4, X_label_iq_train_M4)
```

```
Alpha:  0.1
Subtrain MAE =  1.7441
Subvalid MAE =  2.8631
Time lapsed:  0.008 seconds
```

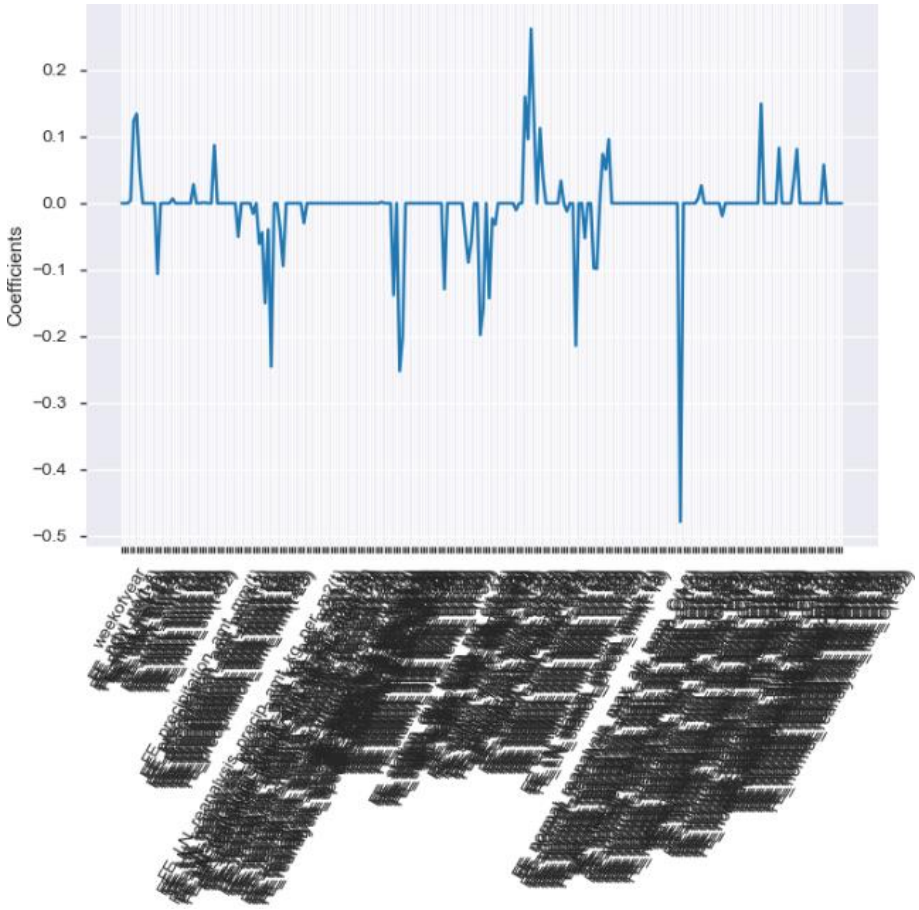


Figure 5.5D: M4 model – LassoBasic1

The Lasso coefficient in Figure 5.5E demonstrated that

- Initial coefficients ranges from (-1.0 to +1.30).
- The magnitude of coefficients decrease as alpha increases.
- All coefficients zerrolized when alpha is larger than 1

```
# Part 5D - Basic Lasso (Simple 70:30 Data Split) Over Different Alphas

# Set alpha range
Alphas2_M4 = np.logspace(-2, 1, 50)

# iq_train, Model 4 - Function call to Basic_Lasso2(Alpha2, X_subtrain, y_subtrain)
Basic_Lasso2(Alphas2_M4, X_iq_subtrain_M4, y_iq_subtrain_M4)
```

```
Time lapsed:  0.5544 seconds
```

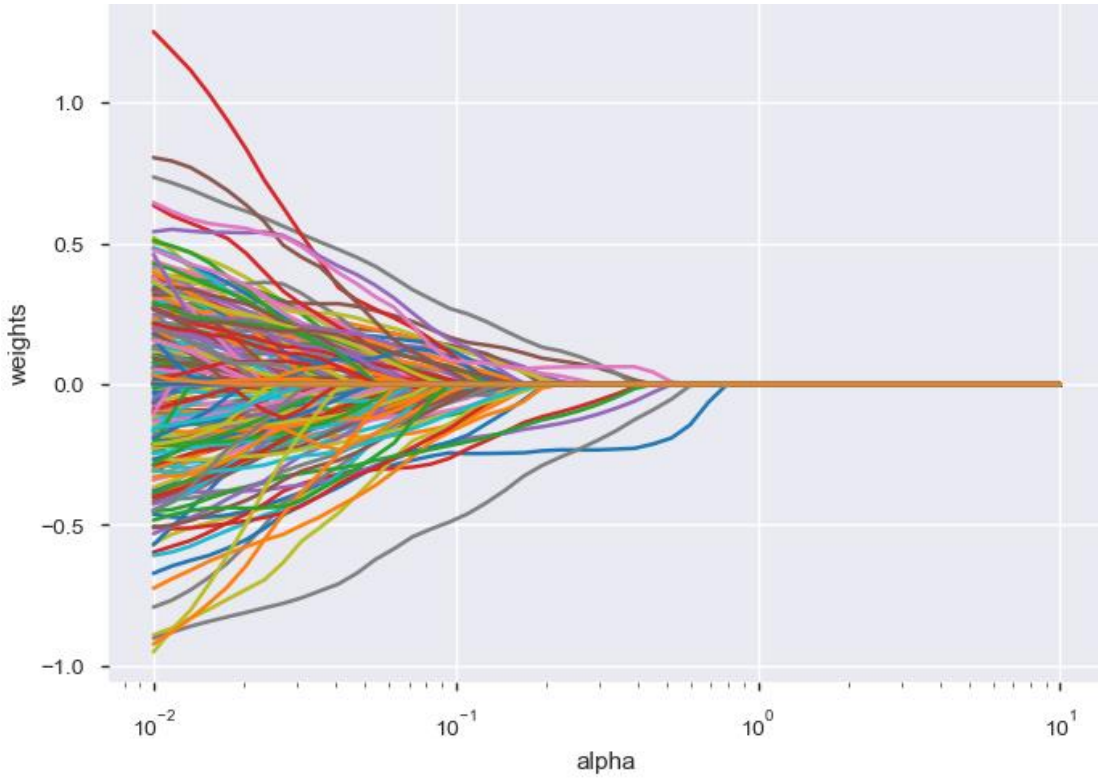


Figure 5.5E: M4 – Lasso coefficient path

Initial grid search in Figure 5.5F found no unique solition negative MAE chart peak and plateaued for alpha => 1.2067926406393288. However, this corresponds to weight = 0 in Figure 5.4E. This will cause model with insignificant coefficients and zero predictions which are not useful at all.

```
# Part 5E - Optimize Lasso Using GridSearchCV with 10-splits from TimeSeriesSplit
Alphas3_M4_init = np.logspace(-2, 1, 50)

# sj_train, Model 2 - Function call to Lasso_GridSearchCV_TimeSeriesSplit(Alphas3, X_subtrain, y_subtrain, X_subvalid, y_subvali
Lasso_iq_M4_init = Lasso_GridSearchCV_TimeSeriesSplit(Alphas3_M4_init, X_iq_subtrain_M4, y_iq_subtrain_M4,
                                                      X_iq_subvalid_M4, y_iq_subvalid_M4)
```

```
Optimized alpha:  1.2067926406393288
Subtrain MAE =  1.7217
Subvalid MAE =  2.3488
Time lapsed:  5.8071  seconds
```

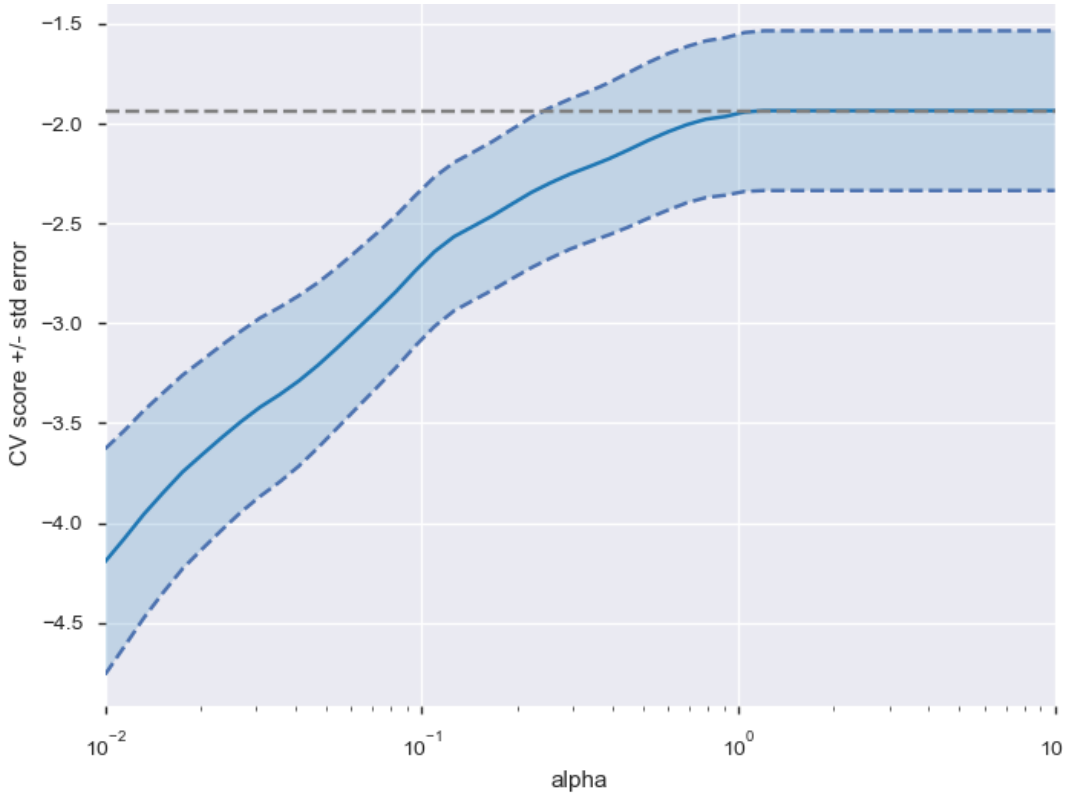


Figure 5.5F: M4 – Initial grid search

Therefore, we need to fine tune the alphas to be the range of 10E-0.75 to 10E-0.35 where there are at least 5 sets of engineered features with non-zero weights. According to Figure 5.4G, unique solution was found at alpha = 0.31622776601683794 and MAE deteriorated from subtrain 1.7741 to subvalid 1.1781.

```
# Choose alpha range where there are about 20 predictors will remain
Alphas3_M4 = np.logspace(-1, -0.5, 100)

# iq_train, Model 4 - Function call to Lasso_GridSearchCV_TimeSeriesSplit(Alphas3, X_subtrain, y_subtrain, X_subvalid, y_subvalid
Lasso_iq_M4 = Lasso_GridSearchCV_TimeSeriesSplit(Alphas3_M4, X_iq_subtrain_M4, y_iq_subtrain_M4,
                                                 X_iq_subvalid_M4, y_iq_subvalid_M4)
```

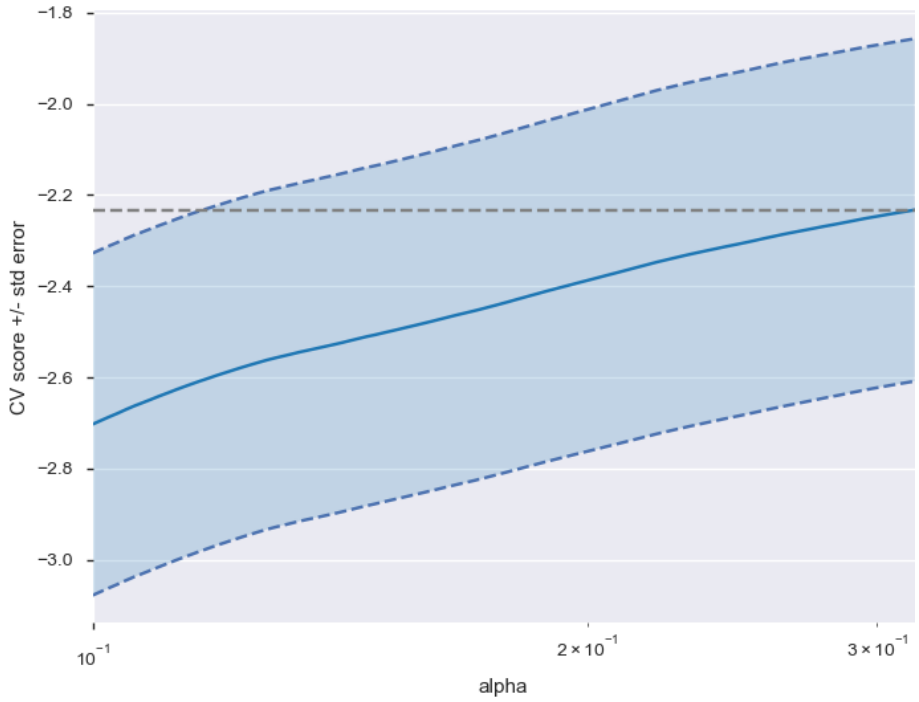


Figure 5.5G: M4 model – Refined grid search

According to Figure 5.5H, 8/264 predictors were found to be significant in predicting Advance4W_normal_total_cases with non-zero coefficients. The top 5 significant predictors are

- FE_LN_reanalysis_precip_amt_kg_per_m2 from 13 weeks ago - the more rainfall, the lesser cases.
- FFE_normal_reanalysis_min_air_temp_C from 13 weeks ago - the higher the minimum temperatures, the more cases.
- FE_station_avg_temp_c of 19-20 weeks ago - the hotter the ground average temperature, the lesser cases
- FE_normal_reanalysis_tdtr_C from 11 weeks ago - the larger the temperature gap, the more cases.
- FE_station_min_temp_c of 0, 2, 23 weeks ago - the higher the minimum ground temperature, the more cases.
- FE_LN_precipitation_amt_mm of 32 weeks ago - the more rainfall, the lesser the cases.

```
# iq_train, Model 4
# Lasso_iq_M4, X_iq_subtrain_M4 - Function call to Lasso_Coef(model, X_subtrain, X_label)
Lasso_Coef_iq_M4 = Lasso_Coef(Lasso_iq_M4, X_iq_subtrain_M4, X_label_iq_train_M4)
```

```
Non-zero coefficient in descending absolute value order:
                                              Coef
FE_LN_reanalysis_precip_amt_kg_per_m2(t-13) -0.231969
FE_normal_reanalysis_min_air_temp_C(t-13)   -0.183084
FE_station_avg_temp_c(t-20)                 -0.096481
FE_normal_reanalysis_tdtr_C(t-11)            0.061924
FE_station_min_temp_c(t-23)                 -0.059650
FE_station_min_temp_c(t-2)                   0.057043
FE_station_min_temp_c(t)                     0.049160
FE_station_avg_temp_c(t-19)                 -0.047247
```

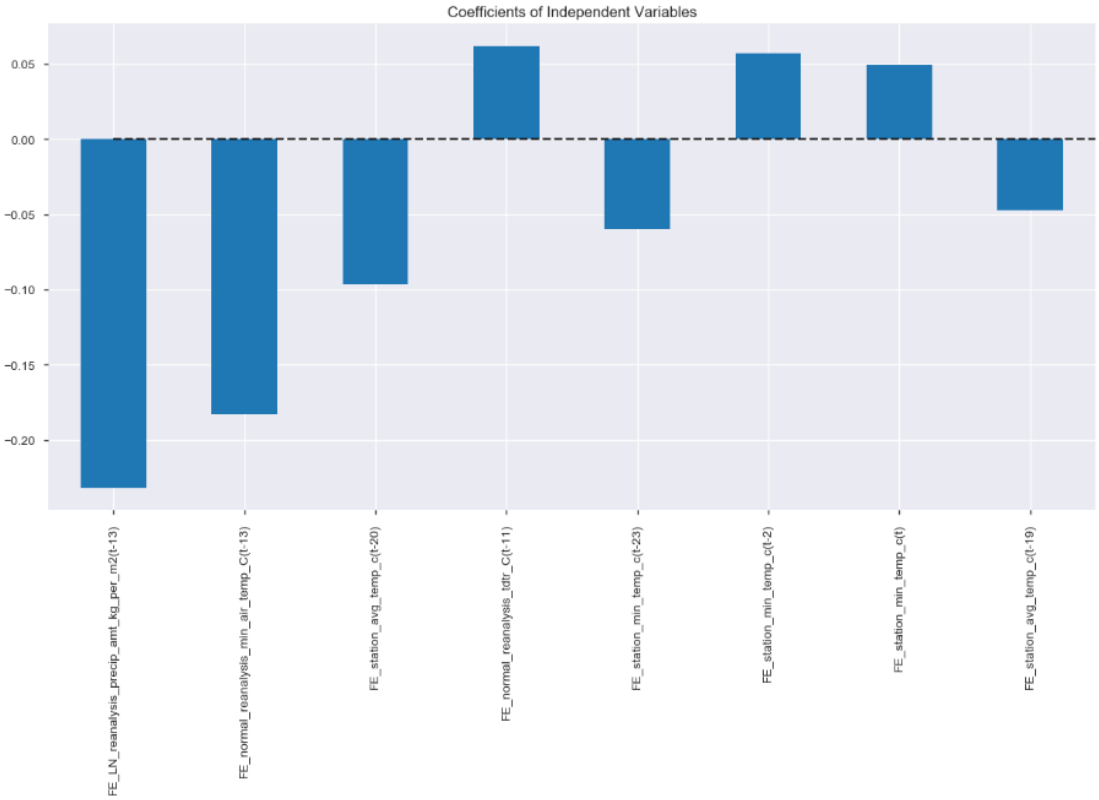


Figure 5.5H: M4 – Lasso coefficients

M4 model's predictions for subtrain and subvalid datasets were matched with the corresponding row indices in Figure 5.5 I-J.

```
# PART 5F - Create Dataframe for Lasso Predictions

# iq_train, Model 4 - Function call to LassoPredict(Lasso_model, X, row_index, DependentVar)
Predict_iq_subtrain_M4 = LassoPredict(Lasso_iq_M4, X_iq_subtrain_M4, iq_index_subtrain_M4, DependentVar_M4)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 338 entries, iq2001-02-12 to iq2007-08-06
Data columns (total 1 columns):
Advance4W_normal_total_cases    338 non-null float64
dtypes: float64(1)
memory usage: 5.3+ KB
None
                     Advance4W_normal_total_cases
City_week_start_date
iq2001-02-12                             0.116850
iq2001-02-19                             0.512289
iq2001-02-26                             0.437540
iq2001-03-05                             0.174631
iq2001-03-12                             0.734268
                     Advance4W_normal_total_cases
City_week_start_date
iq2007-07-09                                  0.0
iq2007-07-16                                  0.0
iq2007-07-23                                  0.0
iq2007-07-30                                  0.0
iq2007-08-06                                  0.0
```

Figure 5.5I: M4 model – predictions for subtrain

```
# iq_train, Model 4 - Function call to LassoPredict(Lasso_model, X, row_index, DependentVar)
Predict_iq_subvalid_M4 = LassoPredict(Lasso_iq_M4, X_iq_subvalid_M4, iq_index_subvalid_M4, DependentVar_M4)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 146 entries, iq2007-08-13 to iq2010-05-28
Data columns (total 1 columns):
Advance4W_normal_total_cases    146 non-null float64
dtypes: float64(1)
memory usage: 2.3+ KB
None
                     Advance4W_normal_total_cases
City_week_start_date
iq2007-08-13                             0.000000
iq2007-08-20                             0.000000
iq2007-08-27                             0.644210
iq2007-09-03                             0.014289
iq2007-09-10                             0.242768
                     Advance4W_normal_total_cases
City_week_start_date
iq2010-04-30                                  0.0
iq2010-05-07                                  0.0
iq2010-05-14                                  0.0
iq2010-05-21                                  0.0
iq2010-05-28                                  0.0
```

Figure 5.5J: M4 model – predictions for subvalid

X and y were prepared from iq_test (Figure 5.4K) before predicting using M4 model (Figure 5.4L). M4 predictions for subtrain, subvalid and test datasets were merged in Figure 5.4M.

```
# Part 5A - Prepare Independent & Dependent Variables

# iq_train, Model M4 - Function call to Prepare_Xy(city, df, AddFE, DependentVar)
X_iq_test_M4, y_iq_test_M4, X_label_iq_test_M4 = Prepare_Xy(city_M4, iq_test, AddFE_M4, DependentVar_M4)
```

```
Number of rows in independent & dependent variables matched after removing NaN from 32 weeks lagged variables.


<class 'pandas.core.frame.DataFrame'>
Index: 160 entries, iq2010-06-04 to iq2013-06-25
Data columns (total 242 columns):
weekofyear                                160 non-null float64
FE_ndvi_ne(t-7)                           160 non-null float64
FE_ndvi_ne(t-8)                           160 non-null float64
FE_ndvi_ne(t-10)                          160 non-null float64
FE_ndvi_ne(t-11)                          160 non-null float64
FE_ndvi_ne(t-31)                          160 non-null float64
FE_ndvi_ne(t-32)                          160 non-null float64
FE_ndvi_sw(t-4)                           160 non-null float64
FE_ndvi_sw(t-5)                           160 non-null float64
FE_ndvi_sw(t-6)                           160 non-null float64
FE_ndvi_sw(t-7)                           160 non-null float64
FE_ndvi_sw(t-8)                           160 non-null float64
```

Figure 5.5K: M4 model – prepare X and y for sj_test

```
# PART 5F - Create Dataframe for Lasso Predictions

# Model 4, X_iq_test - Function call to LassoPredict(Lasso_model, X, row_index)
Predict_iq_test_M4 = LassoPredict(Lasso_iq_M4, X_iq_test_M4, iq_index_test, DependentVar_M4)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 160 entries, iq2010-06-04 to iq2013-06-25
Data columns (total 1 columns):
Advance4W_normal_total_cases    160 non-null float64
dtypes: float64(1)
memory usage: 2.5+ KB
None
                     Advance4W_normal_total_cases
City_week_start_date
iq2010-06-04                                  0.0
iq2010-06-11                                  0.0
iq2010-06-18                                  0.0
iq2010-06-25                                  0.0
iq2010-07-02                                  0.0
                     Advance4W_normal_total_cases
City_week_start_date
iq2013-05-28                              0.13657
iq2013-06-04                              0.00000
iq2013-06-11                              0.00000
iq2013-06-18                              0.00000
iq2013-06-25                              0.00000
```

Figure 5.5L: M4 model – predictions for iq_test

```
# PART 7A - Concatenate Lasso Predictions From y_sj_subtrain_M1, y_sj_subvalid_M1 & y_sj_test_M1

# iq, Model 4 – Function call to Merge_LassoPred (Pred_subtrain, Pred _subvalid, Pred _test, Model_number)
Lasso_prediction_M4 = Merge_LassoPred(Predict_iq_subtrain_M4, Predict_iq_subvalid_M4, Predict_iq_test_M4, 'M4')
```

```
                    Advance4W_normal_Lasso_pred_M4
City_week_start_date
iq2001-02-12                              0.116850
iq2001-02-19                              0.512289
iq2001-02-26                              0.437540
iq2001-03-05                              0.174631
iq2001-03-12                              0.734268


                    Advance4W_normal_Lasso_pred_M4
City_week_start_date
iq2013-05-28                               0.13657
iq2013-06-04                               0.00000
iq2013-06-11                               0.00000
iq2013-06-18                               0.00000
iq2013-06-25                               0.00000
```

Figure 5.5M: M4 model – merge predictions for subtrain, subvalid & test datasets

M4 predictions were merged with actual total_cases (Figure 5.4N) before plotting them in the same chart (Figure 5.5O). M4's predictions are similar to those of M3 whereby the predictions were much lower compared to the actual cases.

```
# iq, Model 4 - Function call to Merge_FEPredict_CityFE(df_FEPredict, df_cityFE)
iq_complete_M4 = Merge_FEPredict_CityFE(Lasso_prediction_M4, iq_completeFE)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 708 entries, iq2000-07-01 to iq2013-06-25
Columns: 267 entries, weekofyear to Advance4W_normal_Lasso_pred_M4
dtypes: float64(263), int32(1), object(3)
memory usage: 1.4+ MB


                    weekofyear week_start_date   ndvi_ne   ndvi_sw  \
City_week_start_date
iq2000-07-01         -0.034978      2000-07-01 -0.875229 -0.226974
iq2000-07-08          0.031377      2000-07-08 -0.580320 -0.291205
iq2000-07-15          0.097732      2000-07-15 -1.073849 -1.608102
iq2000-07-22          0.164088      2000-07-22 -0.446147 -0.770287
iq2000-07-29          0.230443      2000-07-29  0.796587  1.092242

                    precipitation_amt_mm  reanalysis_precip_amt_kg_per_m2  \
City_week_start_date
iq2000-07-01                   -1.103605                            43.19
iq2000-07-08                   -0.102748                            46.00
```

Figure 5.5N: M4 model – merger of predicted and actual total_case

```
# iq, Model 4 - Function call to Plot_Actual_Predict(df_Pred_Features, city, Model_number)
Plot_Actual_Predict(iq_complete_M4, 'Iquitos', 'M4')
```

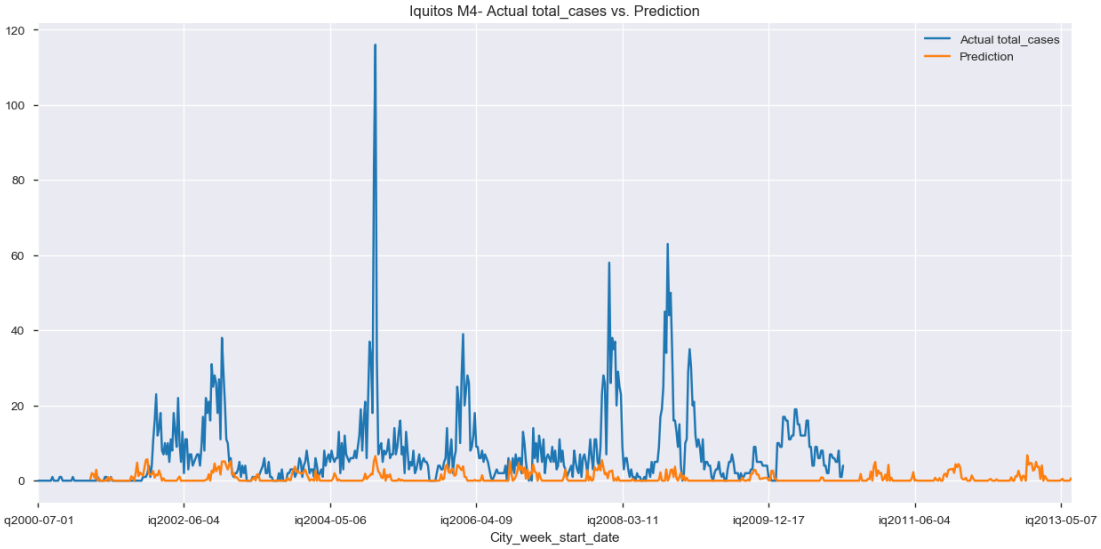


Figure 5.5O: M2 model –predicted vs. actual total_cases

5.6 Model Diagnostics

Please refer to "PART 12 - Visualize Time Series Residual Forecast Errors" in attached Jupyter notebook.

An UDF entitled ResidualPlots creates the following diagnostics plots

- Descriptive Statistics
- Residual Line Plot
- Residual Histogram
- Density Plots
- Residual Q-Q Plot
- Residual Autocorrelation Plot

5.6.1 San Juan M1's model diagnostics

1. Figure 5.6.1A - Descriptive statistics showed a non-zero mean of residual.

2. Figure 5.6.1B - Residual line plot indicated that M1 did not capture the peaks during outbreak.

3. Figure 5.6.1C - Residual histogram illustrated a right-skewed distribution with some large outliers

4. Figure 5.6.1D - Density plots recorded similar observation as Figure 5.6C.

5. Figure 5.6.1E - Residual Q-Q plot presented a non-normal distribution with right-skewed, outliers.

6. Figure 5.6.1F - Residual autocorrelation plot did not mimic a white noise pattern since there are significant autocorrelation about 150, 200, 350, 550, 650 lags, which could be some cyclical effect.

```
# sj, Model 1 - Function call to ResidualPlots(df_complete, city_FullName, Model_number)
ResidualPlots(sj_complete_M1, city_FullName = 'San Juan', Model_number = 'M1')
```

```
San Juan M1 - Descriptive statistics of Actual, Predicted, Residual:
```

	total_cases	Lasso_pred_M1	residual
count	900.000000	900.000000	900.000000
mean	34.562222	8.762872	25.799351
std	52.225058	11.303351	49.291543
min	0.000000	0.000000	-38.266196
25%	9.000000	0.000000	4.896005
50%	19.000000	1.850986	12.854289
75%	37.000000	16.703147	28.185090
max	461.000000	48.316922	430.996858

Figure 5.6.1A: M1 – descriptive statistics of actual, prediction and residual

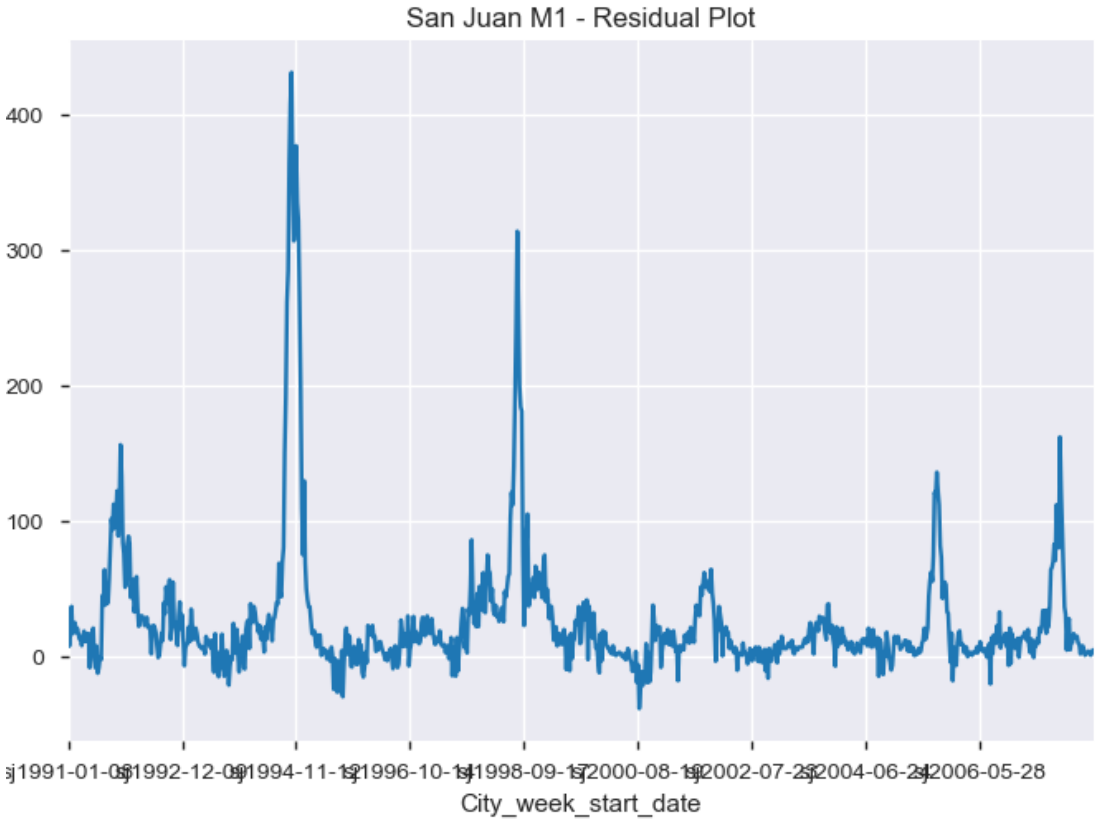


Figure 5.6.1B: M1 – residual line plot

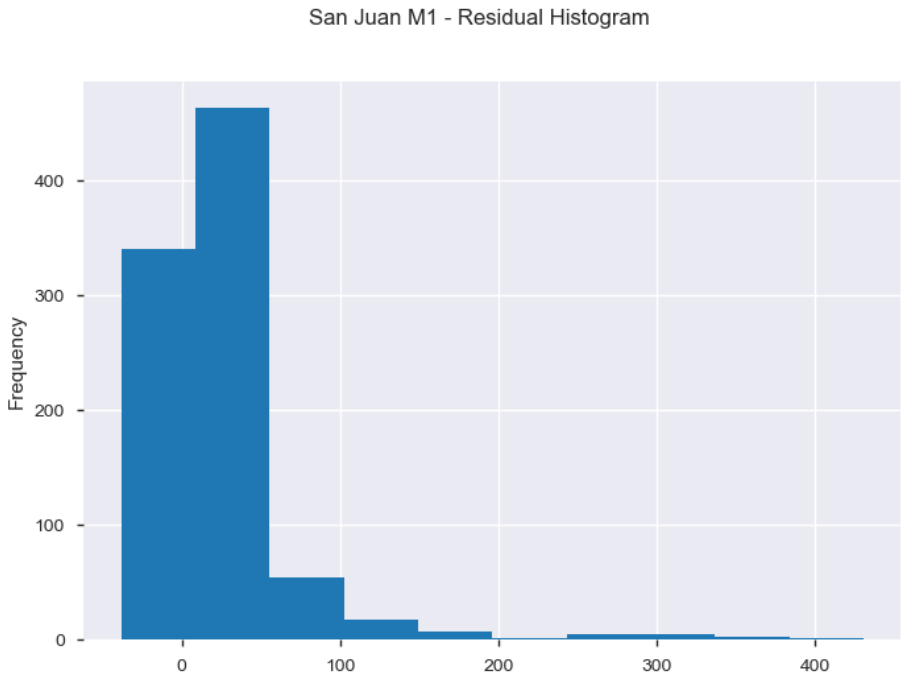


Figure 5.6.1C: M1 – residual histogram

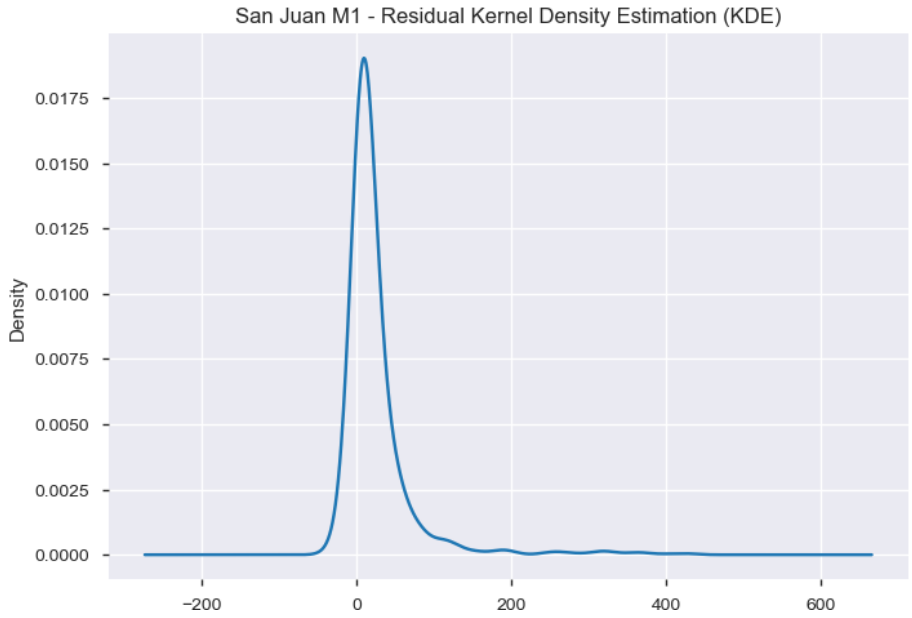


Figure 5.6.1D: M1 – residual KDE

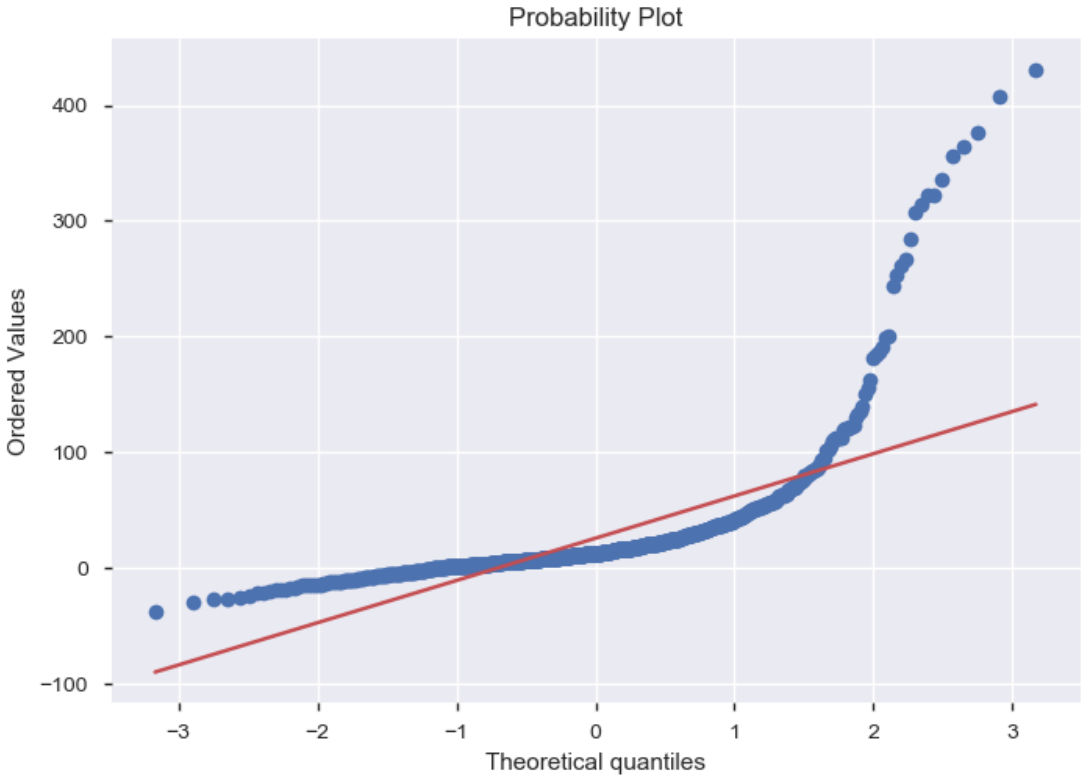


Figure 5.6.1E: M1 – QQ plot

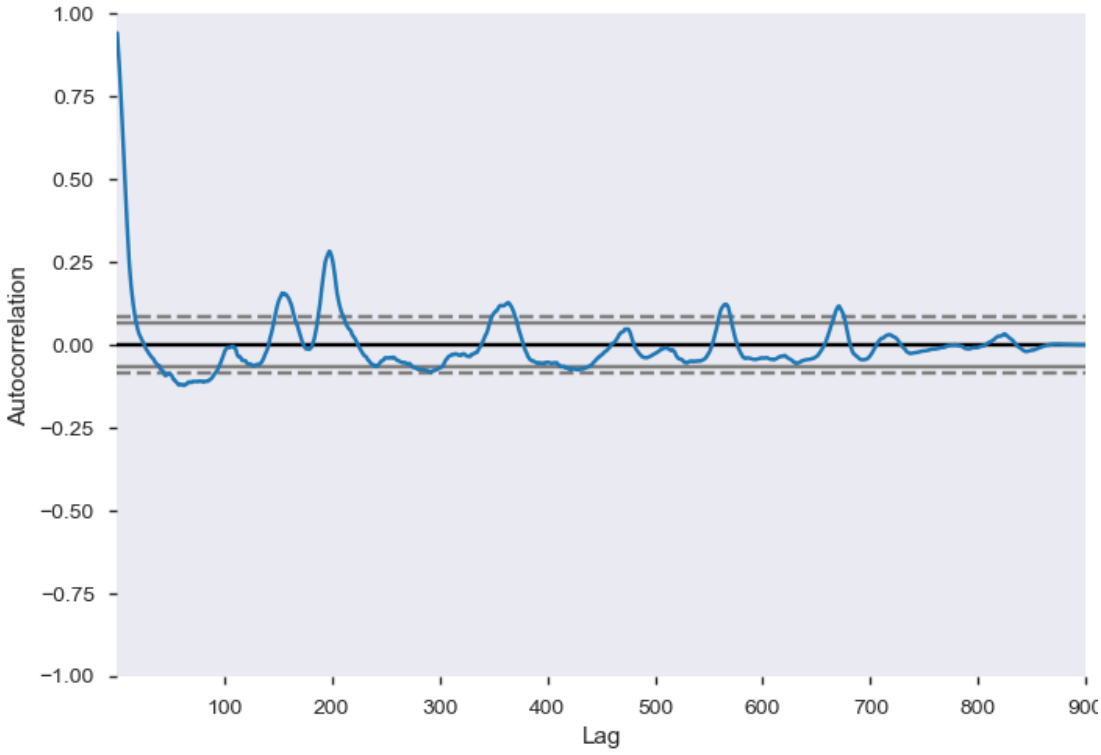


Figure 5.6.1F: M1 – residual ACF plot

5.6.2 San Juan M2's model diagnostics

1. Figure 5.6.2A - Descriptive statistics showed a non-zero mean of residual.

2. Figure 5.6.2B - Residual line plot indicated that M1 did not capture the peaks during outbreak.

3. Figure 5.6.2C - Residual histogram illustrated a right-skewed distribution with some large outliers

4. Figure 5.6.2D - Density plots recorded similar observation as Figure 5.6.2C.

5. Figure 5.6.2E - Residual Q-Q plot presented a non-normal distribution with right-skewed, outliers.

6. Figure 5.6.2F - Residual autocorrelation plot did not mimic a white noise pattern since there are significant autocorrelation about 150, 200, 350, 550, 700 lags, which could be some cyclical effect.

```
# sj, Model 2 - Function call to ResidualPlots(df_complete, city_FullName, Model_number)
ResidualPlots(sj_complete_M2, city_FullName = 'San Juan', Model_number = 'M2')
```

```
San Juan M2 - Descriptive statistics of Actual, Predicted, Residual:


       total_cases  Lasso_pred_M2    residual
count   932.000000     932.000000  932.000000
mean     34.849785       6.996110   27.853675
std      52.031687      10.345720   48.066886
min       0.000000       0.000000  -24.212002
25%       9.000000       0.000000    5.879164
50%      19.000000       0.000000   14.000000
75%      37.000000      12.075978   30.449852
max     461.000000      41.939781  420.922586
```

Figure 5.6.2A: M2 – descriptive statistics of actual, prediction and residual

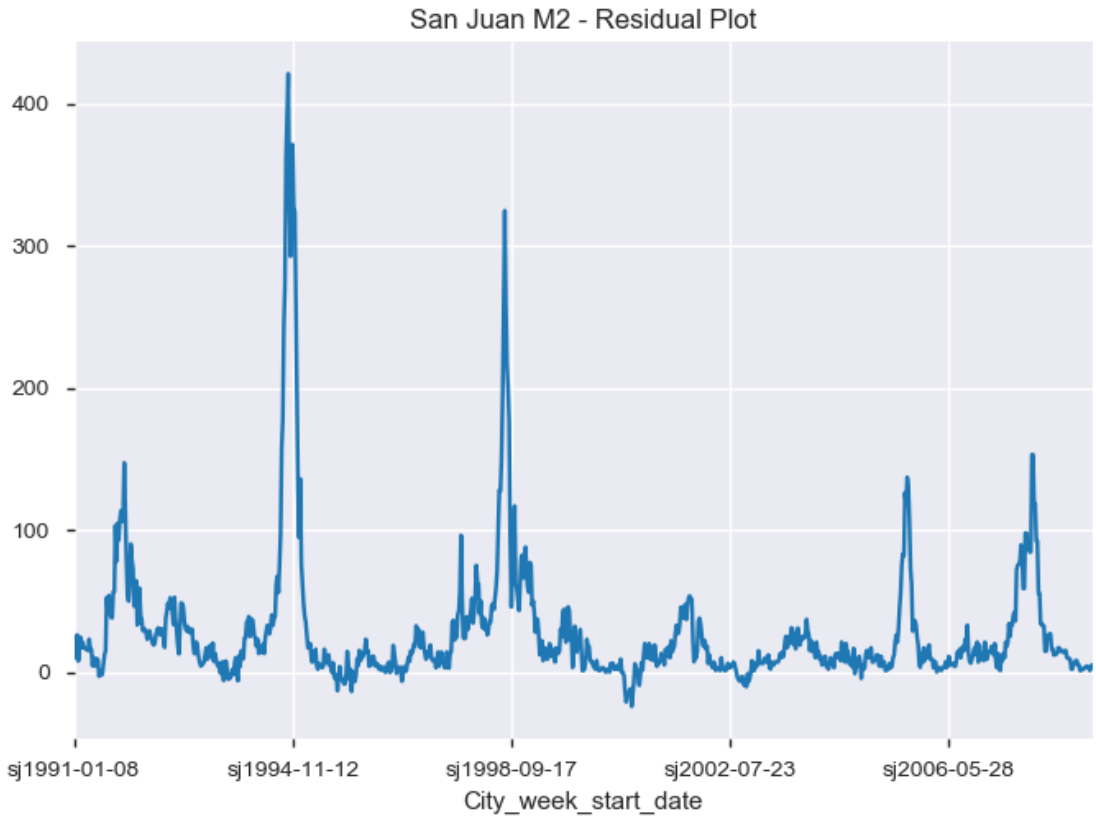


Figure 5.6.2B: M2 – residual line plot

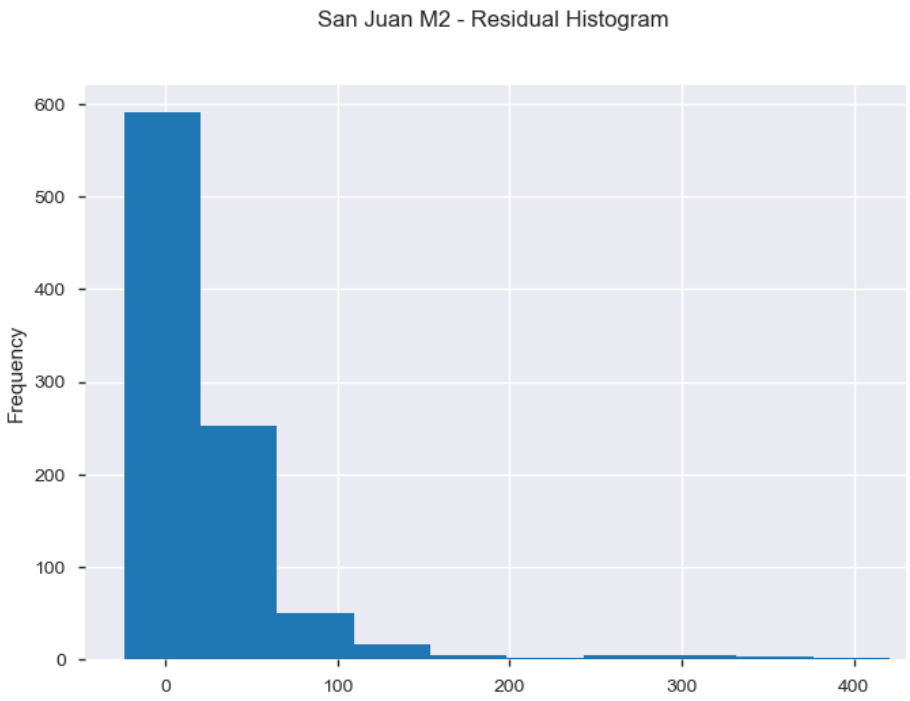


Figure 5.6.2C: M2 – residual histogram

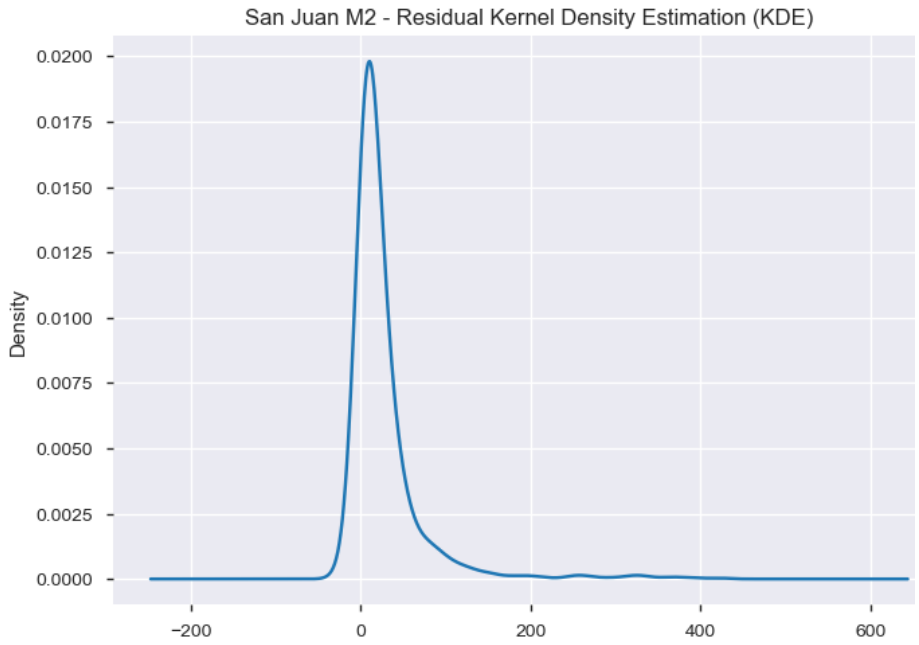


Figure 5.6.2D: M2 – residual KDE

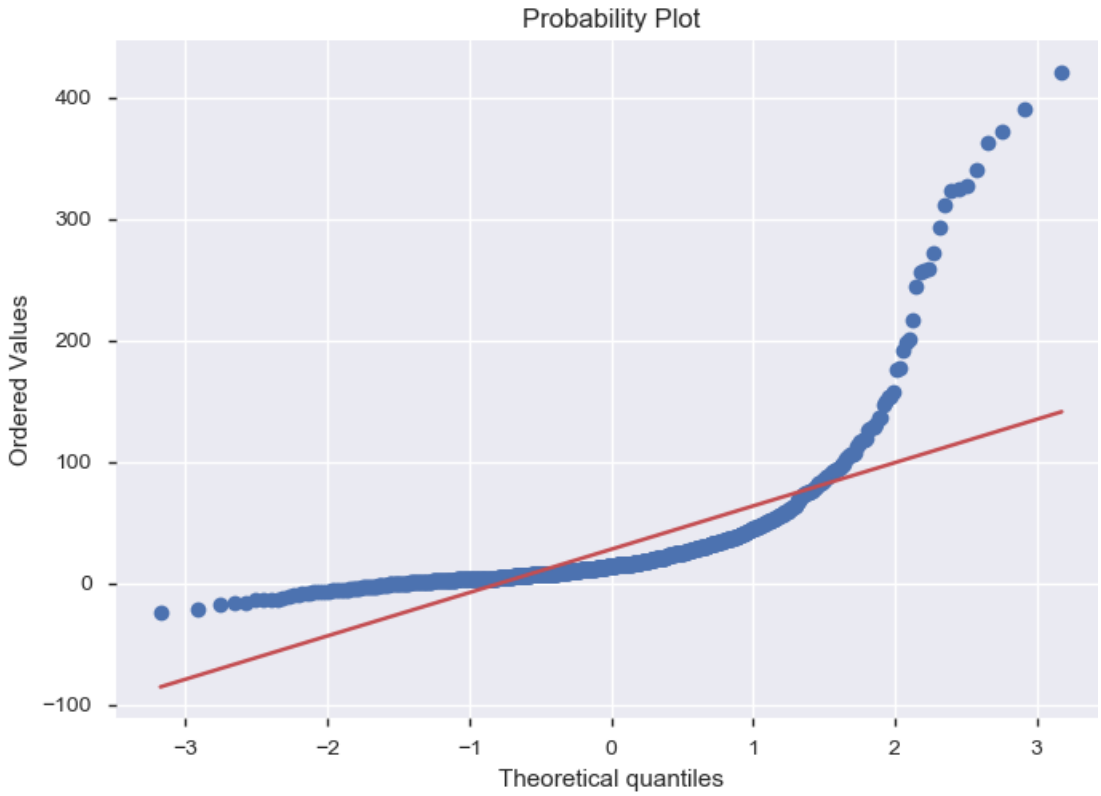


Figure 5.6.2E: M2 – QQ plot

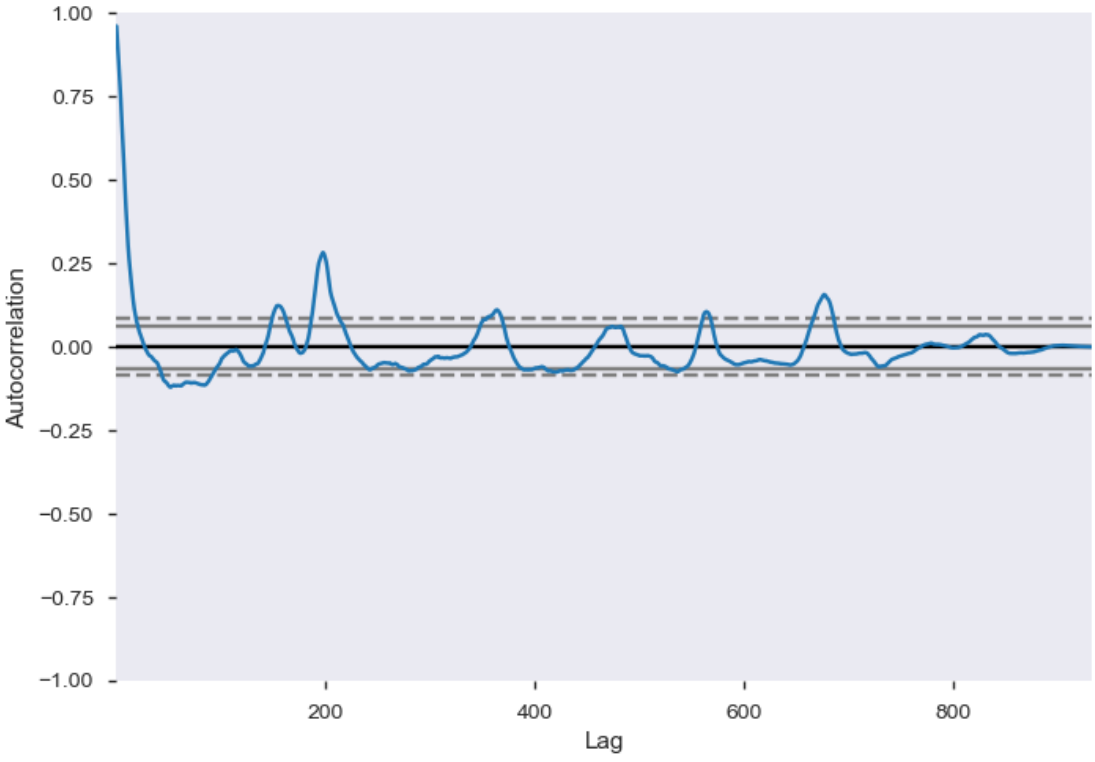


Figure 5.6.2F: M2 – residual ACF plot

5.6.3 Iquitos M3 model diagnostics

1. Figure 5.6.3A - Descriptive statistics showed a non-zero mean of residual.

2. Figure 5.6.3B - Residual line plot indicated that M1 did not capture the peaks during outbreak.

3. Figure 5.6.3C - Residual histogram illustrated a right-skewed distribution but the outliers are not as large as San Juan

4. Figure 5.6.3D - Density plots recorded similar observation as Figure 5.6.3C.

5. Figure 5.6.3E - Residual Q-Q plot presented a non-normal distribution with right-skewed, outliers.

6. Figure 5.6.3F - Residual autocorrelation plot did not mimic a white noise pattern since there are significant autocorrelation about 100, 200 lags. There could be some cyclical effect which is less prominent than San Juan.

```
# iq, Model 3 - Function call to ResidualPlots(df_complete, city_FullName, Model_number)
ResidualPlots(iq_complete_M3, city_FullName = 'Iquitos', Model_number = 'M3')
```

```
Iquitos M3 - Descriptive statistics of Actual, Predicted, Residual:

       total_cases  Lasso_pred_M3    residual
count   484.000000     484.000000  484.000000
mean      8.119835       0.758109    7.361726
std      10.957952       1.050269   10.792765
min       0.000000       0.000000   -3.177887
25%       2.000000       0.000000    1.000000
50%       5.000000       0.093355    4.349431
75%      10.000000       1.337228    9.000000
max     116.000000       4.915136  112.335162
```

Figure 5.6.3A: M3 – descriptive statistics of actual, prediction and residual

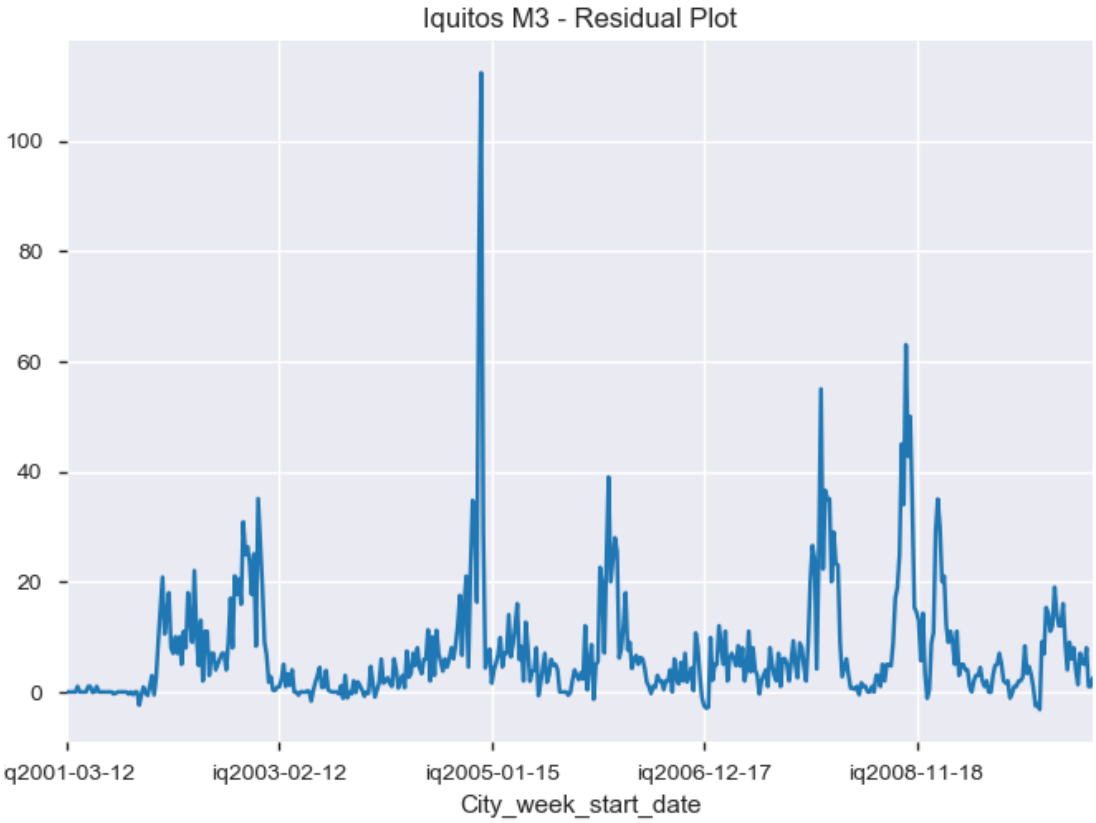


Figure 5.6.3B: M3 – residual line plot

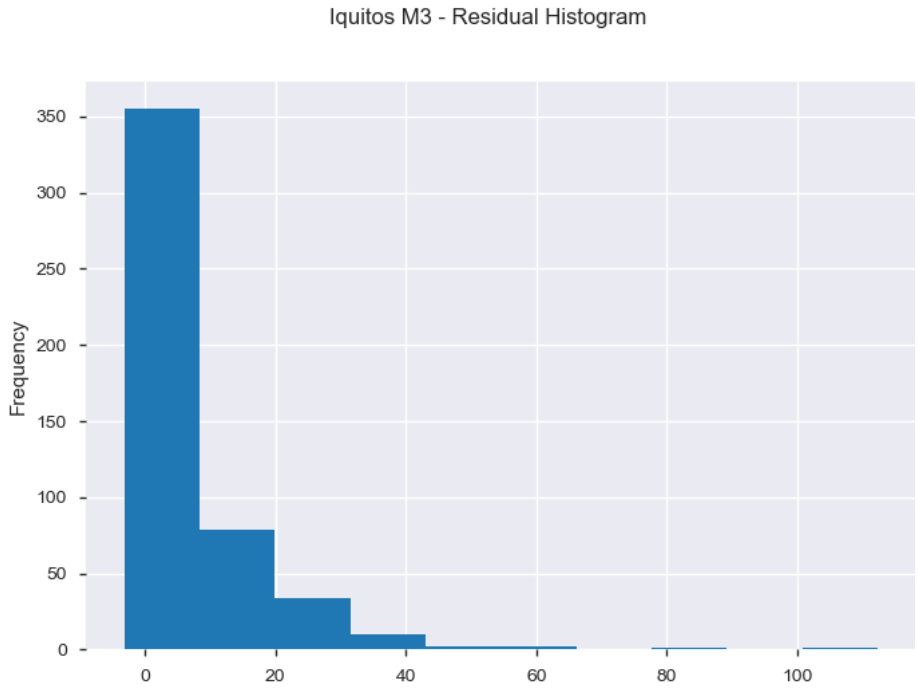


Figure 5.6.3C: M3 – residual histogram

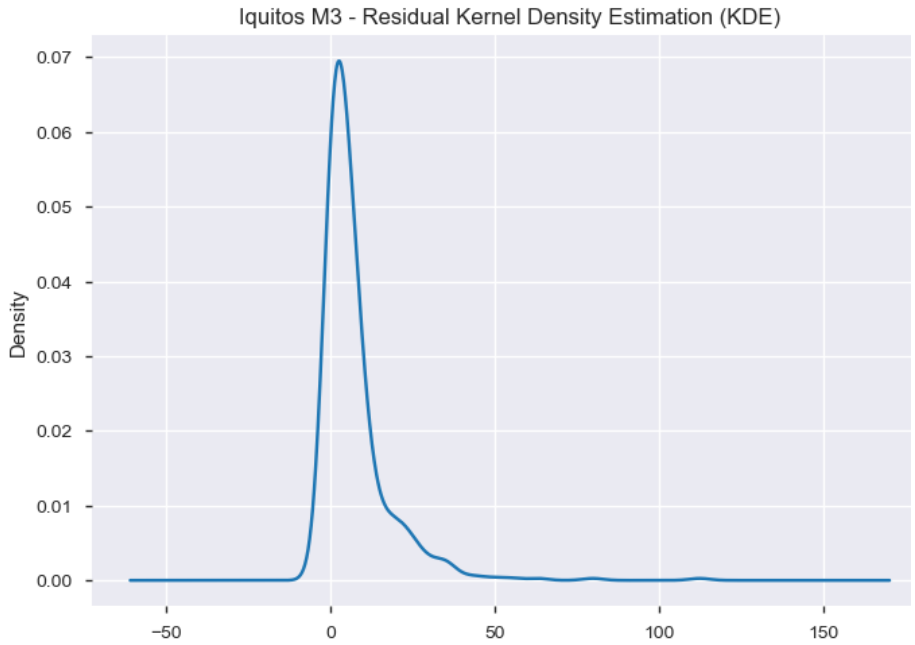


Figure 5.6.3D: M3 – residual KDE

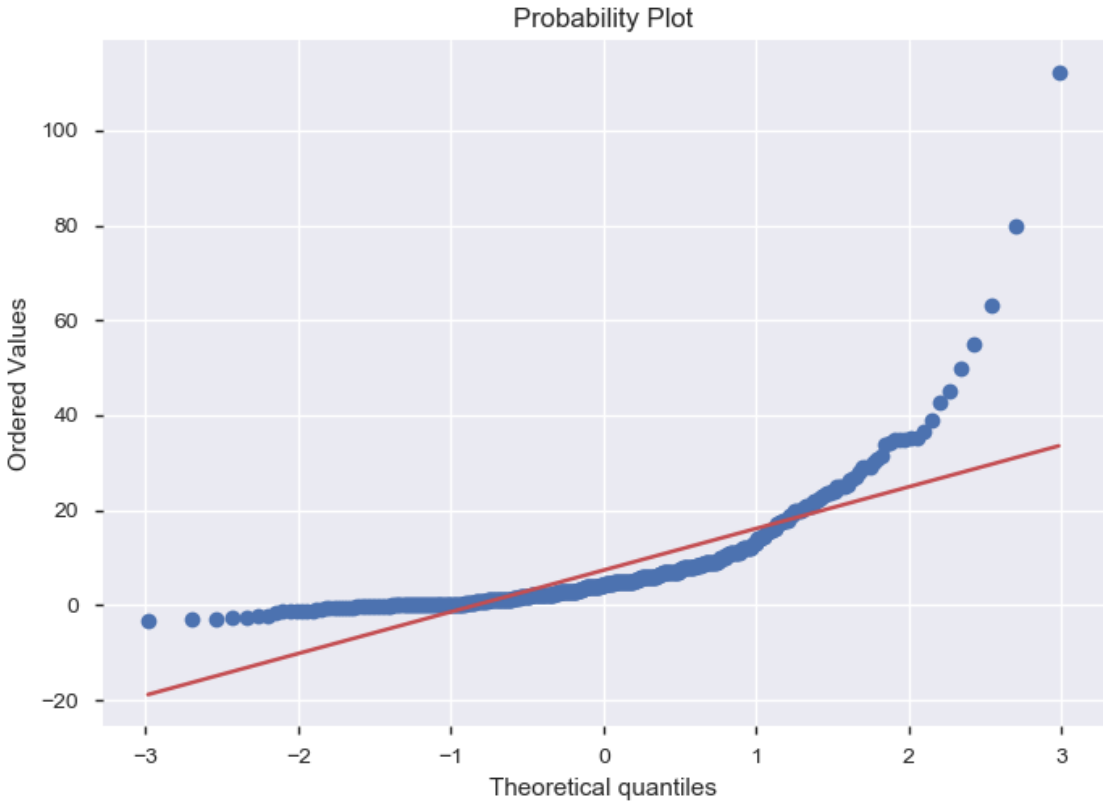


Figure 5.6.3E: M3 – QQ plot

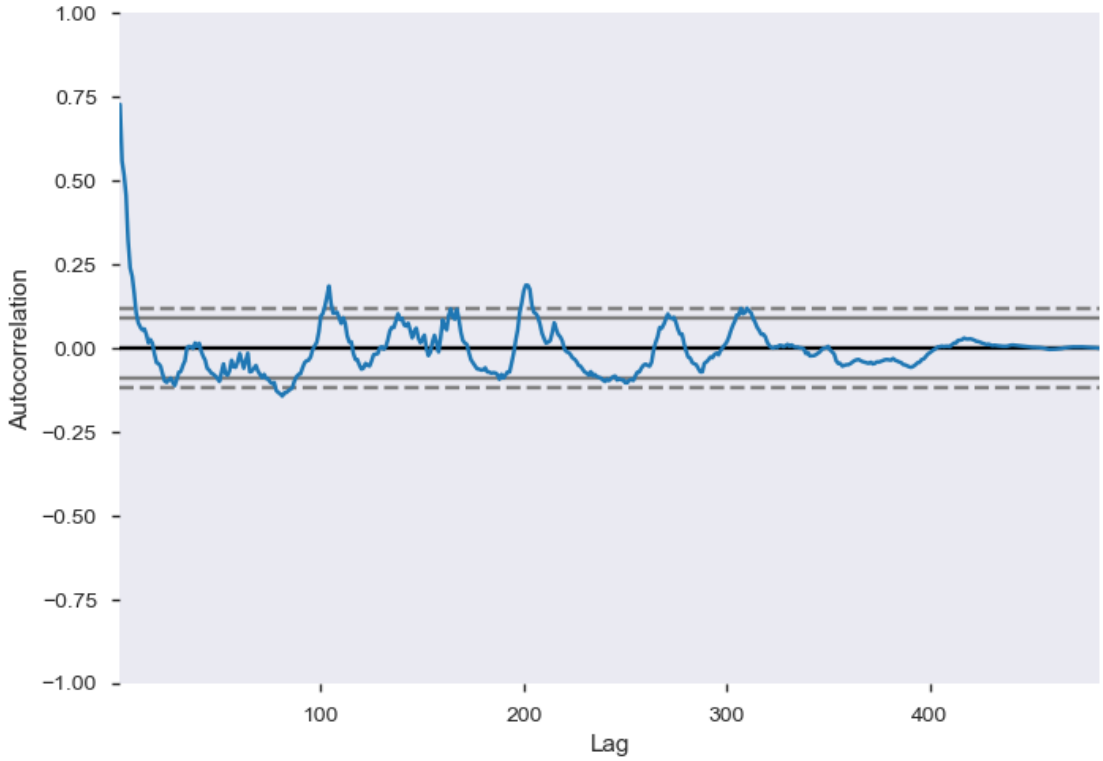


Figure 5.6.3F: M3 – residual ACF plot

5.6.4 Iquitos M3 model diagnostics

1. Figure 5.6.4A - Descriptive statistics showed a non-zero mean of residual.

2. Figure 5.6.4B - Residual line plot indicated that M1 did not capture the peaks during outbreak.

3. Figure 5.6.4C - Residual histogram illustrated a right-skewed distribution with comparable outliers as M3.

4. Figure 5.6.4D - Density plots recorded similar observation as Figure 5.6.4C.

5. Figure 5.6.4E - Residual Q-Q plot presented a non-normal distribution with right-skewed, outliers.

6. Figure 5.6.4F - Residual autocorrelation plot did not mimic a white noise pattern since there is significant autocorrelation about 200 lags. There could be some cyclical effect which is less prominent than M3.

```
# iq, Model 4 - Function call to ResidualPlots(df_complete, city_FullName, Model_number)
ResidualPlots(iq_complete_M4, city_FullName = 'Iquitos', Model_number = 'M4')
```

```
Iquitos M4 - Descriptive statistics of Actual, Predicted, Residual:


       total_cases  Lasso_pred_M4    residual
count   516.000000     516.000000  516.000000
mean      8.089147       0.756685    7.332462
std      10.690155       1.286080   10.315572
min       0.000000       0.000000   -4.784538
25%       2.000000       0.000000    1.165768
50%       5.000000       0.000000    5.000000
75%      10.000000       1.066330    9.000000
max     116.000000       6.541859  109.458141
```

Figure 5.6.4A: M4 – descriptive statistics of actual, prediction and residual

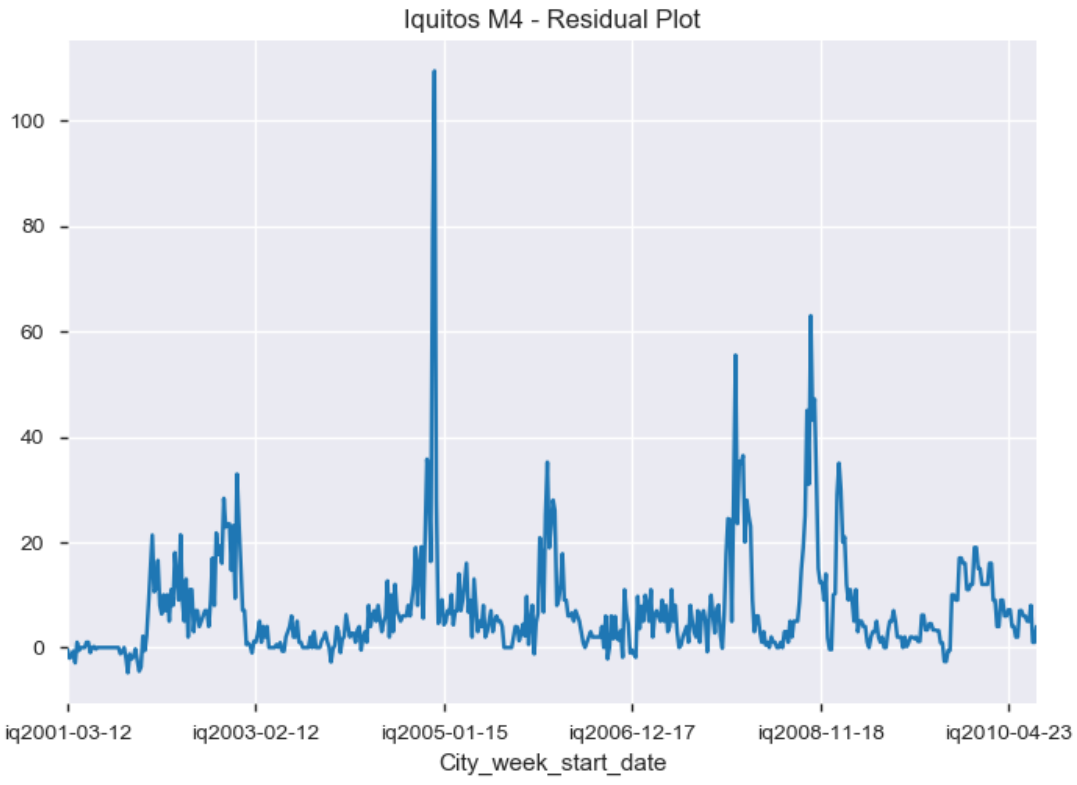


Figure 5.6.4B: M4 – residual line plot

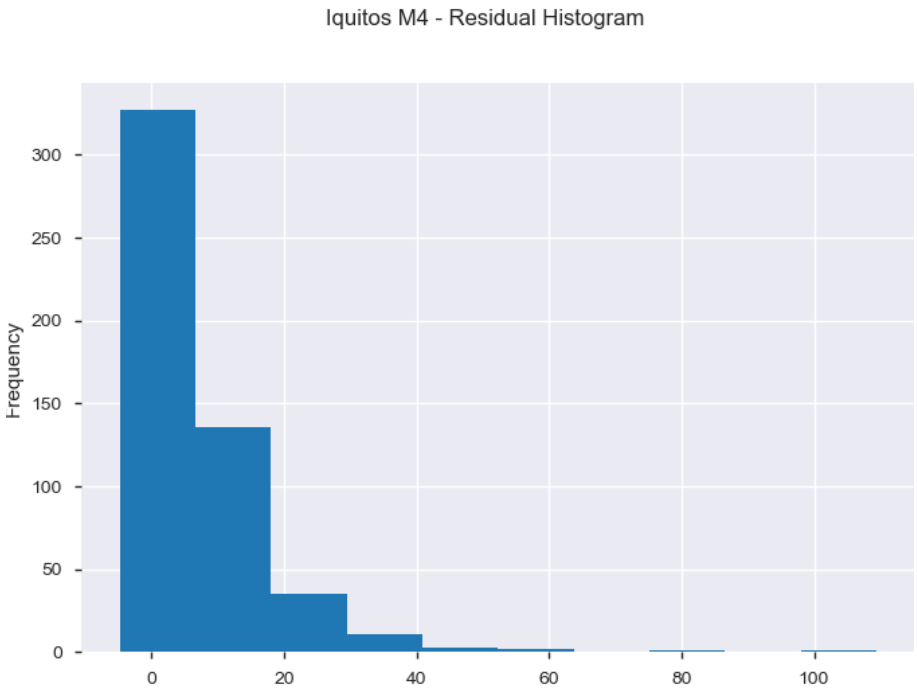


Figure 5.6.4C: M4 – residual histogram

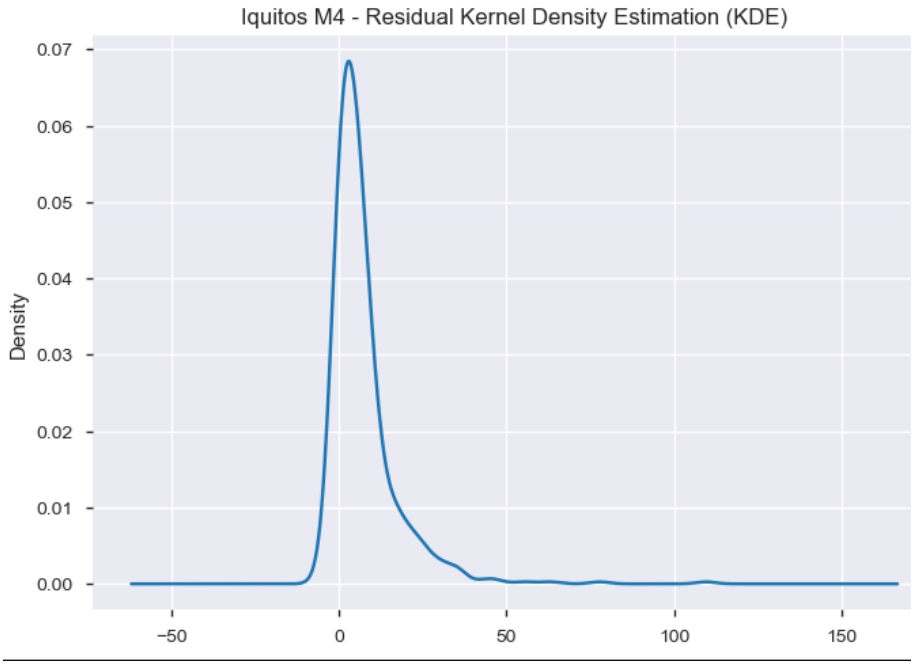


Figure 5.6.4D: M3 – residual KDE

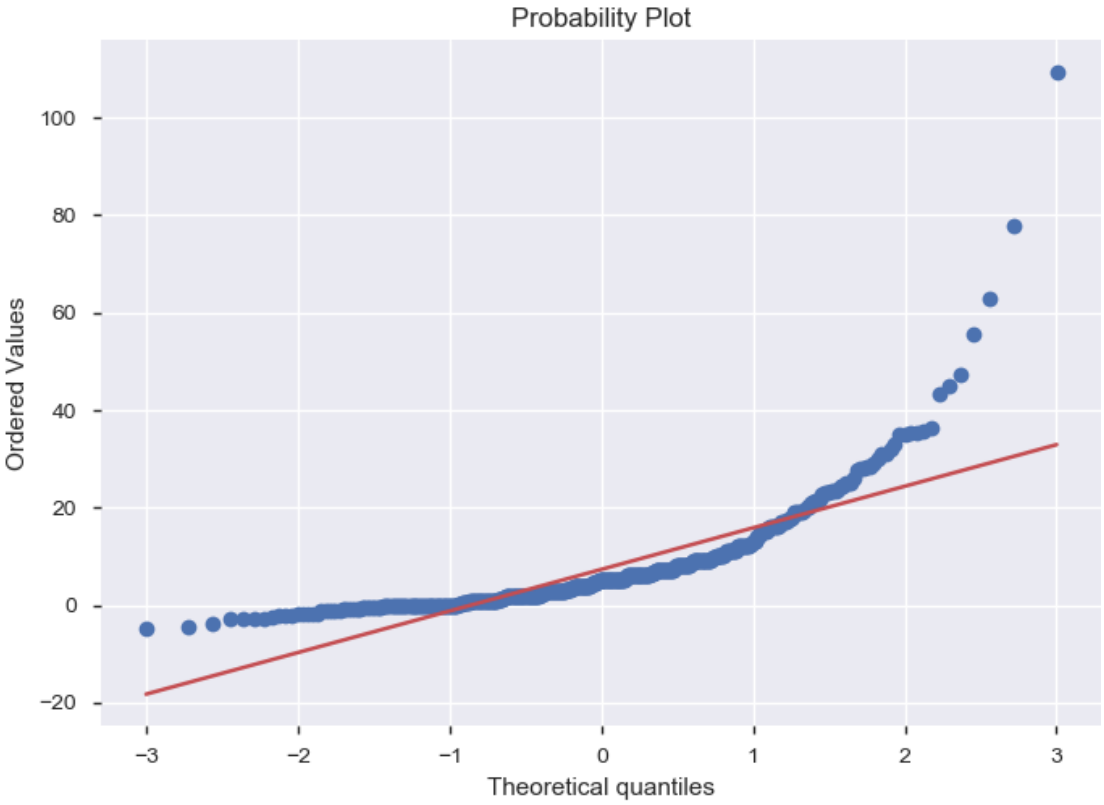


Figure 5.6.4E: M4 – QQ plot

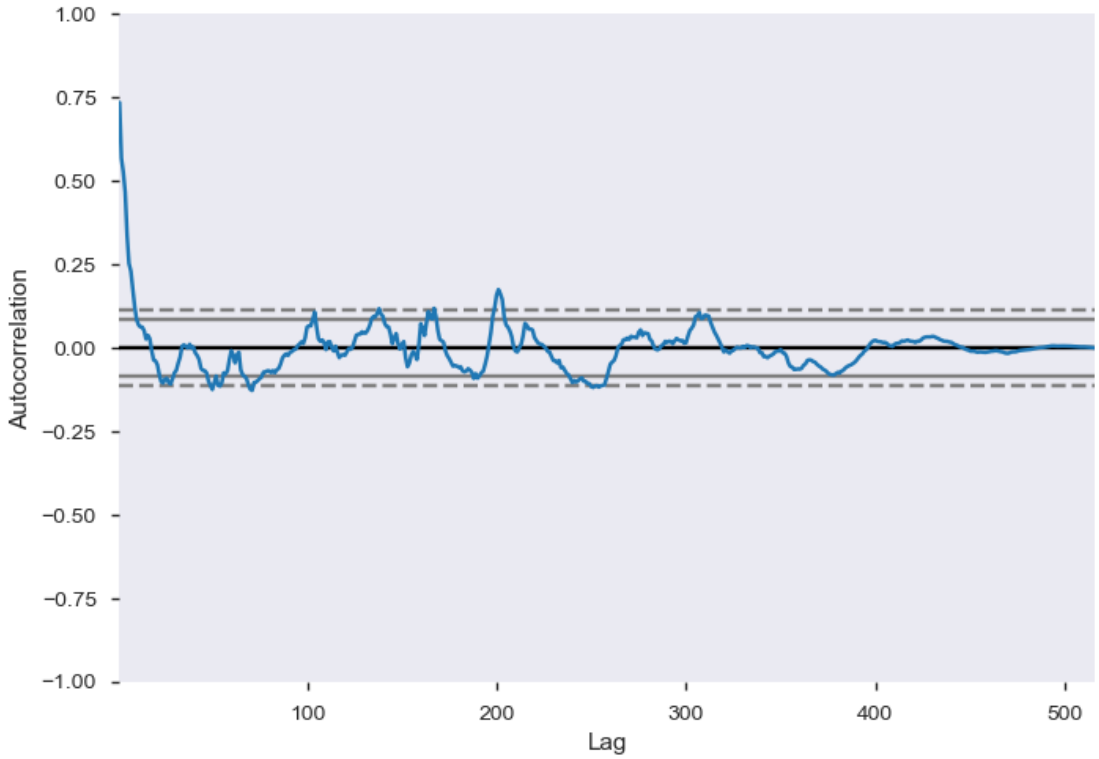


Figure 5.6.4F: M4 – residual ACF plot

CHAPTER 6
DISCUSSION AND CONCLUSIONS

6.1 Discussion

6.1.1 Google Trend

Please refer to "Appendix 1 – Google Trends API" on access application and project information.

According to Table 6.1, this research could be the first attempt to extract city level epidemic data for a developing country.

Table 6.1 – How Google Trends was utilized in epidemic research

References	Disease	Country	Data level	Data extraction method
Lu et al. (2018a)	Flu	US	City	API
Lu et al. (2018b)	Flu	US	State	API
Zou, Lampos & Cox (2018)	Flu	US	State	API
		UK	Country	
Yang et al. (2017)	Dengue	Mexico, Brazil, Thailand, Singapore and Taiwan	Country	Website
Teng et al. (2017)	Zika	Worldwide	Wordwide	Website
Sampri, Mavragani & Tsagarakis (2016)	Asthma, Lyme disease, Melanoma, COPD, Salmonella	US	State	Website

The data extract for San Juan and Iquitos was not successful as both cities do not have ISO 3166-2 ticker that Google Trends API requires to identify region, state and city. San Juan is located in Puerto Rico which has only country level ISO ticker. While Iquitos has no ISO 3166-2 ticker but it is location within the Loreto region which carries "PER-LOR" as ISO regional ticker.

Please refer to "peru_region_loreto - Comparison multi window.xlsx" for attempts to proxy Iquitos using regional level data from Loreto whereby

- Data prior to 2004 cannot be extracted

- Data for the remaining training period from 1-Jan-2004 until 1-Jul-2010 are all zeros for daily & weekly basis.
- Aggregation from daily to weekly and monthly did not tally, even taking the sampling effect into consideration.

Therefore, no signal can be drawn from Google Trends to be used as predictor.

6.1.2 Implausible values from MICE imputation

The single imputation of MICE produced a negative precipitation value in Figure 5.4M which caused NaN for logarithmic transformation. In addition to the possibility of churning out implausible values, MICE based on stochastic regression face problems with heteroscedastic data. The new Predictive Mean Matching (PMM) Imputation overcome the two aforementioned weaknesses of stochastic regression based MICE (Statistical Programming, 2019).

6.1.3 Cross validation for time series analysis

The traditional K-fold Cross Validation (CV) assumed each observation is independent of the other. However, this is not the case with the temporal dependent (Cochrane, 2018) dengue cases and climatic data. Hence, the Time Series cross-validator (scikit-learn developers, 2018h) which is represented by Figure 6.1A was used in this research.

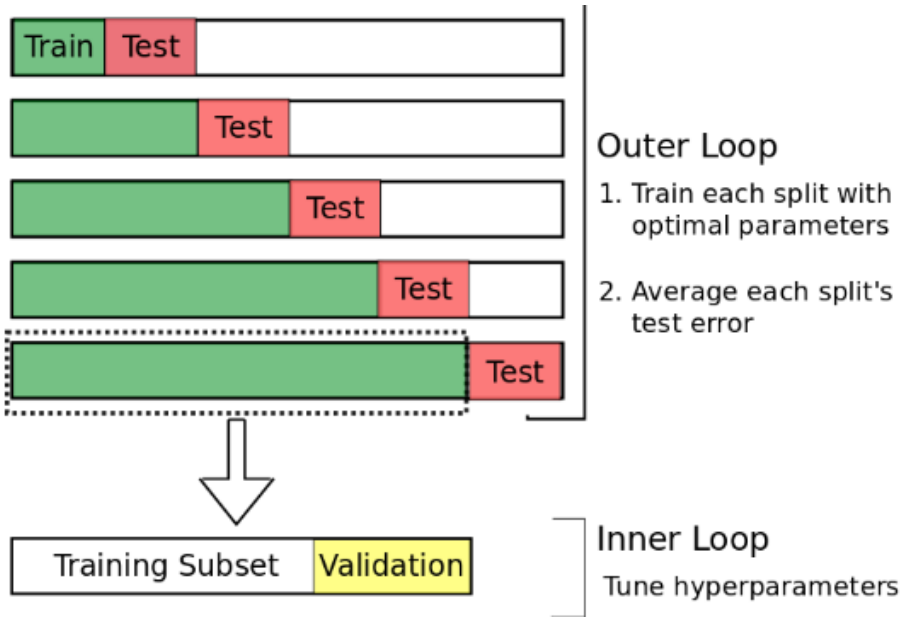


Figure 6.1A: Nested cross validation (Cochrane, 2018)

A more sophisticated "evaluation on a rolling forecasting origin" in Figure 6.1B could also be adopted in the future.

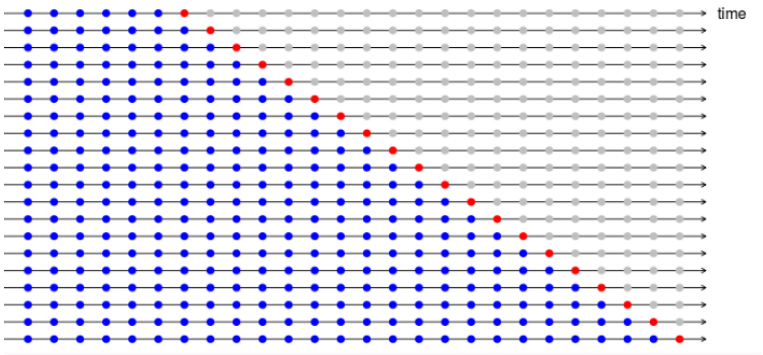


Figure 6.1B: Evaluation on a rolling forecasting origin where the blue and red dots are training and test sets respectively (Hyndman, 2016)

6.1.4 Modelling for San Juan

Over-shrinkage (Musoro et al., 2016) occurred whereby the minimum MAE was found when optimal alpha (lambda in R's terminology) is larger than 1 in Figure 5.2.5A and Figure 5.3F for M1 and M2 respectively. This could be caused by the lack of strong linear relationship whereby most of the weeks have very low dengue cases excepted during outbreak periods. Hence, the grid search were refined to work in the range whereby there are at least 5 sets of non-zero coefficient in the Lasso coefficient path in Figure 5.2.4 and Figure 5.3E for M1 and M2 respectively.

Model M1 is favoured over M2 due to

- Table 6.1 - smaller overall MAE
- Figure 5.2.11 – better able to mimic the peaks

This means that the extensive engineered features in M2 did not add much value in predicting the dengue cases in San Juan, compared to the simpler M1. However, both M1-M2

- Did not capture the outbreaks based on time series plots (Figure 5.2.11, Figure 5.3O).
- Have significant cyclical components based on residual ACF plots (Figure 5.6.1F, Figure 5.6.2F).

Table 6.2 – M1 vs. M2 for San Juan

Data set	Period	M1			M2		
		MAE	Alpha	Significant predictors	MAE	Alpha	Significant predictors
subtrain	10-Dec-1990 to 15-Jan-2003	1.3401	0.1	• station_max_temp_C • normal_reanalysis_tdtr_C – • weekofyear • ndvi_nw • LN_renalaysis_precip_amt_kg_per_m2	1.6195	0.56234	• FE_normal_reanalysis_max_air_temp_C from 28-32 weeks ago. • FE_station_diur_temp_rng_c from 23, 26-32 weeks ago. • FE_ndvi_nw of 28, 30-32 weeks ago. • FE_station_min_temp_c of 6 weeks ago. • FE_LN_precipitation_amt_mm of 32 weeks ago.
subtest	22-Jan-2003 to 25-Mar-2008	1.1104			1.1781		

6.1.5 Modelling for Iquitos

Over-shrinkage occurred for M4 model (Figure 5.5F) whereby minimum MAE was found for alpha larger than one. However, both M3 (Figure 5.4E) and M4 (Figure 5.5) have non-unique alphas whereby MAE charts do not have unique peak.

According to Tibshirani (2012), the non-unique Lasso solutions could occur when

- p > n with p = number of variables p and n = number of observations.
- Discrete predictors were used.
- Post-processing was carried out on continuous predictors.

Taylor (2012) asserted that if X^TX of formula (5) is invertible, the no unique Lasso solution can be found.

$$\hat{\beta} = (X^T X)^{-1} X^T Y \qquad (5)$$

p = 242 and n = 484 for Iquitos yields p / n = 0.5 the first condition is not fulfilled. weekofyear is the only discrete variable used in both cities but it was standardized, hence nullify the second condition. No further processing was none on the continuous variables in both cities, which invalidate the third condition. The remaining invertible possibility of Iquitos' X^TX needs to be further investigated and is outside the scope of this research.

Similar to San Juan's over-shrinkage problem in, Iquitos' grid search were refined to work in range whereby there are at least 5 sets of non-zero coefficient in the Lasso coefficient path in Figure 5.4D and Figure 5.5E for M3 and M4 respectively. Model M4 is favoured over M3 due to

- Table 6.1 - smaller MAE in subtest
- Figure 5.6.4F: better able to capture the cyclical component

This means that the extensive engineered features in M4 added value in predicting the dengue cases in Iquitos, compared to the simpler M3. However, both M3-M4 did not capture the outbreaks based on time series plots (Figure 5.4X, Figure 5.5O).

Table 6.2 – M3 vs. M4 for Iquitos

Dataset	Period	M3			M4		
		MAE	Alpha	Significant predictors	MAE	Alpha	Significant predictors
subtrain	12-Feb-2001 to 6-Aug-2007	1.7303	0.1	• station_min_temp_c • weekofyear LN_station_precip_mm • ndvi_ne • LN_reanalysis_precip_amt_kg_per_m2 station_avg_temp_c	1.7741	0.3162	• FE_LN_reanalysis_precip_amt_kg_per_m2 from 13 weeks ago. • FFE_normal_reanalysis_min_air_temp_C from 13 weeks ago. • FE_station_avg_temp_c of 19-20 weeks ago. • FE_normal_reanalysis_tdtr_C from 11 weeks ago. • FE_station_min_temp_c of 0, 2, 23 weeks ago. • FE_LN_precipita

							tion_amt_mm of 32 weeks ago.
subtest	13-Aug-2007 to 28-May-2010	2.4149			1.1781		

6.1.6 Comparison of San Juan vs. Iquitos

Temperature is the key and consistent predictor in both cities, where the hotter the temperature, the more dengue cases.

However, the different effects were noted for

- The gap between the lowest and highest temperature of the day, whereby normal_reanalysis_tdtr_C has negative coefficient in San Juan but FE_normal_reanalysis_tdtr_C has positive coefficients.
- Precipitation whereby LN_renalaysis_precip_amt_kg_per_m2 has positive coefficient in San Juan but Iquitos' FE_LN_reanalysis_precip_amt_kg_per_m2 from 13 weeks ago and FE_LN_precipitation_amt_mm of 32 weeks ago have negative coefficients.

San Juan is more sensitive to year end and vegetation effects compared to Iquitos. On the other hand, the lagged climatic variables effects are more profound in Iquitos. The lengths of significant climatic lags are found to be consistent with those of Sougata, Acebedo and Chua (2017).

Table 6.4 – The significant predictors of San Juan vs. Iquitos

San Juan – M1	Iquitos – M4
• station_max_temp_C - the hotter, the more cases. • normal_reanalysis_tdtr_C - the bigger the gap between maximum & minimum temperatures, the lesser cases. • weekofyear - cases increase toward year end. • ndvi_nw - the more	• FE_LN_reanalysis_precip_amt_kg_per_m2 from 13 weeks ago - the more rainfall, the lesser cases. • FFE_normal_reanalysis_min_air_temp_C from 13 weeks ago - the higher the minimum temperatures, the more cases. • FE_station_avg_temp_c of 19-20 weeks ago - the hotter the ground average temperature, the lesser cases. • FE_normal_reanalysis_tdtr_C from 11 weeks ago -

vegetation on the north-west direction, the more cases. • LN_renalaysis_precip_amt_kg_per_m2 - the more rainfall, the more cases	the larger the temperature gap, the more cases. • FE_station_min_temp_c of 0, 2, 23 weeks ago - the higher the minimum ground temperature, the more cases. • FE_LN_precipitation_amt_mm of 32 weeks ago - the more rainfall, the lesser the cases.

6.1.7 Benchmarking against other nowcasting work

Two studies which nowforecasted dengue are

- Freeze, Erraguntla and Verma (2018) which forecast 1-week and 4-weeks ahead, based on the same DrivenData – DengAI competition.
- Shi et al. (2016) which forecast week 1 to week 12 ahead for Singapore.

Freeze, Erraguntla and Verma (2018) used non-linear terms of square and cube total_cases and weekly percentage change of total_cases which were found to be significant in nowcasting dengue. This research does not involve the derivation of total_cases as this information is not made available in the test dataset. The authors combined both cities in the same MLR, SVM, RF and Boosting models while this research created separate Lasso models for each city. The significant "region" indicator in Freeze, Erraguntla and Verma's model support the separate modelling of two cities.

Shi et al. (2016) built an ensemble of 12 Lasso sub-models instead of a single Lasso model. The authors applied logarithmic transformation of the total_cases in the form of log (1 + normalized case). This has the potential to reduce the right-skewed residuals faced in this research since Table 4.2 recorded improved correlation when target variable underwent logarithmic transformation. The authors also included Breeding Percentage (BP), which measured the dominance of Ae. Aegypti among the breeding sites. This is a crucial predictor as different serotypes dominated dengue outbreaks at different times, as noted in Iquitos (The International Federation of Red Cross and Red Crescent Societies, 2011b). While NOOA provided the serotype breakdown in the 2015 competition, such information is only limited to the train dataset, hence unable to be utilized to forecast the test dataset.

This research along with Freeze, Erraguntla and Verma failed to predict the severe outbreak for Iquitos in 2010.

6.2 Conclusions

This is the first recorded attempt to use city level Google Trend data to forecast dengue. However, both San Juan and Iquitos do not have the corresponding ISO 3166-2 ticker to enable city-level data extraction. Feature engineering added value to Iquitos but not for San Juan nowcasting model. Temperature is the key and consistent predictor in both cities, where the hotter the temperature, the more dengue cases. San Juan is more sensitive to year end and vegetation effects while the lagged climatic variables effects are more profound in Iquitos. However, the different effects were noted for the two cities, namely the gap between the lowest and highest temperature of the day and precipitation.

Predictive Mean Matching (PMM) could be used to overcome the negative imputed values, followed by Bootstrapping and MI-LASSO (Musuro et al. 2014) to further validate the multiple imputations used in Lasso model. Current Lasso models could be complemented with other submodels in an ensemble framework to capture cyclical components of climatic variables.

6.3 Future Recommendations

This research is limited by the exclusion of extreme weather and non-linear effect.

Predictive Mean Matching (PMM) could be used to overcome the negative imputed values, followed by Bootstrapping and MI-LASSO (Musuro et al. 2014) to further validate the multiple imputations used in Lasso model. Both San Juan and Iquitos should obtain the city level ISO 3166-2 ticker to enable the evaluation of Google Trends as crowd-sourced signal in nowcasting dengue. Current Lasso models could be complemented with other submodels in an ensemble framework to capture cyclical components of climatic variables.

REFERENCES

Akinfaderin, W. (2017) *Missing Data Conundrum: Exploration and Imputation Techniques*. [Online]. Available from: https://medium.com/ibm-data-science-experience/missing-data-conundrum-exploration-and-imputation-techniques-9f40abe0fd87. [Accessed: 10/12/2018].

Augspurger, T. (2016) *Modern Pandas (Part 7): Timeseries*. [Online]. Available from: https://tomaugspurger.github.io/modern-7-timeseries. [Accessed: 10/12/2018].

Azencott, C.A. (2015) *Lab 2: Introduction to scikit-learn (Part 2).* [Online]. Available from: http://cazencott.info/dotclear/public/lectures/ma2823_2015/scikit-learn-2.pdf. [Accessed: 10/12/2018].

Bliman, P.A., Codeco, C. and Coelho, F. (2016) *From Detection to Forecasting: Big Data and Models in Epidemiology, the Example of Dengue in Rio de Janeiro* [Online]. June 2016. Available from: https://project.inria.fr/digitalforhealth/files/2016/06/PABlimanvf.pdf. [Accessed: 13/06/2018].

Brownlee, J. (2016) *How To Resample and Interpolate Your Time Series Data With Python*. [Online]. Available from: https://machinelearningmastery.com/resample-interpolate-time-series-data-python/. [Accessed: 08/12/2018].

Brownlee, J. (2017) *A Gentle Introduction to Autocorrelation and Partial Autocorrelation*. [Online]. Available from: https://machinelearningmastery.com/gentle-introduction-autocorrelation-Partial-autocorrelation/. [Accessed: 08/12/2018].

Buczak, A.L., Baugher, B., Moniz, L.J., Bagley, T., Babin, S.M. and Guven, E. (2018) Ensemble method for dengue prediction. *PLOS One*. 13(1). [Online]. Available from: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0189988. [Accessed: 08/07/2018].

CBS Interactive Inc. (2018) *Puerto Rico's population falls by 130,000 people in a year*. [Online]. Available from: https://www.cbsnews.com/news/puerto-ricos-population-falls-by-130000-people-in-a-year/. [Accessed: 23/04/2019].

Chen Y., Chu C.W., Chen, M.I.C. and Cook, A.R. (2018) The utility of LASSO-based models for real time forecasts of endemic infectious diseases: A cross country comparison. *Journal of Biomedical Informatics.* 81. p. 16–30.

Cheepsattayakorn, A. and Cheepsattayakorn, R. (2018) Climate Changes and Human Infectious Diseases. *EC Microbiology.* 14.6. p. 299-311.

Center for Disease Control and Prevention. (2012) *Symptoms and What To Do If You Think You Have Dengue*. [Online]. Available from: https://www.cdc.gov/dengue/symptoms/index.html. [Accessed: 08/07/2018].

Cochrane, C. (2018) *Time Series Nested Cross-Validation*. [Online]. Available from: https://towardsdatascience.com/time-series-nested-cross-validation-76adba623eb9. [Accessed: 23/04/2019].

Comprehensive R Archive Network (CRAN) (2018) *nowcasting: Nowcast Analysis and Create Real-Time Data Basis.* [Online] Available from: https://cran.r-project.org/web/packages/nowcasting/index.html. [Accessed: 09/11/2018].

Country Reports. (2018) *Peru Languages*. [Online]. Available from: http://www.countryreports.org/country/Peru/language.htm. [Accessed: 11/07/2018].

Das, B. (2016) *Missing Data and Imputation*. [Online]. Available from: https://www.naaccr.org/Webinars/Missing%20Data%20and%20Imputation.pdf. [Accessed: 10/12/2018].

DataCamp. (2018) *Autocorrelation Function*. [Online]. Available from: https://s3.amazonaws.com/assets.datacamp.com/production/course_4267/slides/chapter2.pdf. [Accessed: 10/12/2018].

DrivenData. (2018a) *DengAI: Predicting Disease Spread - Challenge Summary*. [Online]. Available from:

https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/page/80/. [Accessed: 09/12/2018].

DrivenData. (2018b) *DengAI: Predicting Disease Spread - Problem description*. [Online]. Available from: https://www.drivendata.org/competitions/44/dengai-predicting-disease-spread/page/82/. [Accessed: 09/12/2018].

El-Metwally, A. (2015) Google Search Trend of Dengue fever in developing Countries in 2013-2014: An Internet-Based Analysis. *Journal of Health Informatics in Developing Countries*. [Online]. 9(1). DOI: 10.1016/j.ijmedinf.2017.05.003. Available from: http://www.jhidc.org/index.php/jhidc/article/viewFile/131/173 [Accessed: 13/06/2018].

European Central Bank. (2013) *Working Paper Series No. 1564 - Now-casting and the real-time data flow*. Online]. July 2013. Available from: https://www.ecb.europa.eu/pub/pdf/scpwps/ecbwp1564.pdf. [Accessed: 09/11/2018].

Freeze, F., Erraguntla, M. and Verma, A. (2018) *Data Integration and Predictive Analysis System for Disease Prophylaxis: Incorporating Dengue Fever Forecasts*. In Proceedings of the 51st Hawaii International Conference on System Sciences. Hawaii, USA: HICSS. p. 913-922.

Google. (2018) *Thank you for stopping by*. [Online]. Available from: http://www.google.org/flutrends/about/. [Accessed: 09/07/2018].

Google. (2019) *Google Trends API Request form*. [Online] Available from: https://docs.google.com/forms/d/e/1FAIpQLSenHdGiGl1YF-7rVDDmmulN8R-ra9MnGLLs7gIIaAX9VHPdPg/viewform [Accessed: 11/04/2019].

Google News Initiative. (2018a) *Google Trends: See what's trending across Google Search, Google News and YouTube*. [Online]. Available from: https://storage.googleapis.com/gweb-news-initiative-training.appspot.com/upload/GO802_NewsInitiativeLessons_Fundamentals-L03-GoogleTrends.pdf. [Accessed: 10/11/2018].

Google News Initiative. (2018b) *Google Trends: Understanding the data*. [Online]. Available from: https://storage.googleapis.com/gweb-news-initiative-training.appspot.com/upload/GO802_NewsInitiativeLessons_Fundamentals-L04-GoogleTrends.pdf. [Accessed: 10/11/2018].

Hyndman, R.J. (2016) *Cross-validation for time series*. Online]. Available from: https://robjhyndman.com/hyndsight/tscv/. [Accessed: 23/04/2019].

International Society for Disease Surveillance (ISDS). (2015) *Analytic Solutions for Real-Time Biosurveillance Advancing the Utility of Infectious Disease Modeling for Public Health Practice*. [Online]. October 2015. Available from: https://knowledge-repository.s3.amazonaws.com/reports/UseCase_2015_07_Forecast%20Modeling_FinalReport.pdf. [Accessed: 09/07/2018].

Jain, S. (2017) *A comprehensive beginners guide for Linear, Ridge and Lasso Regression.* [Online] Available from: https://www.analyticsvidhya.com/blog/2017/06/a-comprehensive-guide-for-linear-ridge-and-lasso-regression/. [Accessed: 21/07/2015].

Johnson, L.R., Gramacy, R.B., Cohen, J., Mordecai, E., Murdock, C., Rohr, J., Ryan, S.J., Stewart-Ibarra, A.M. and Weikel, D. (2018) Phenomenological forecasting of disease incidence using heteroskedastic Gaussian processes: a dengue case study. *Annals of Applied Statistics*. 12(1). p. 27-66.

Kane, D. (2015) *Data Science - Part XII - Ridge Regression, LASSO, and Elastic Nets*. [Online] Available from: https://www.slideshare.net/DerekKane/data-science-Part-xii-ridge-regression-lasso-and-elastic-nets. [Accessed: 21/07/2018].

Khan Academy. (2018) *Exponential and logistic growth in populations*. [Online] Available from: https://www.khanacademy.org/science/biology/ecology/population-growth-and-regulation/v/exponential-and-logistic-growth-in-populations. [Accessed: 10/12/2018].

Koehrsen, W. (2018) *Histograms and Density Plots in Python*. [Online] Available from: https://towardsdatascience.com/histograms-and-density-plots-in-python-f6bda88f5ac0. [Accessed: 10/12/2018].

Laureano-Rosario, A.E.,Duncan, A.P., Mendez-Lazaro, P.A., Garcia-Rejon, J.E., Gomez-Carro, S., Farfan-Ale, J., Savic, D.A. and Muller-Karger, F.E. (2018) Application of Artificial Neural Networks for Dengue Fever Outbreak Predictions in the Northwest Coast of Yucatan, Mexico and San Juan, Puerto Rico. *Tropical Medicine and Infectious Disease.* 3(1). DOI: https://doi.org/10.3390/tropicalmed3010005.

Lee, K.Y., Chung, N. and Hwang, S.T. (2016) Application of an artificial neural network (ANN) model for predicting mosquito abundances in urban areas. *Ecological Informatics journal*. 36. p. 172–180.

Lee, J.S., Carabali, M., Lim, J.K., Herrera, V.M., Park, I.Y., Villar, L. and Farlow, A. (2017) Early warning signal for dengue outbreaks and identification of high risk areas for dengue fever in Colombia using climate and non-climate datasets. *BMC Infectious Diseases*. 17:480. DOI: 10.1186/s12879-017-2577-4. [Online]. Available from: https://bmcinfectdis.biomedcentral.com/articles/10.1186/s12879-017-2577-4. Accessed: 10/07/2018.

Lu, F.S., Hattab, M.W., Clemente, L. and Santillana, M. (2018a) *Improved state-level influenza activity nowcasting in the United States leveraging Internet-based data sources and network approaches via ARGONet*. [Online] Jun 2018. Available from: https://www.biorxiv.org/content/biorxiv/early/2018/06/14/344580.full.pdf [Accessed: 23/04/2019].

Lu, F.S., Hou, S., Baltrusaitis, K., Shah, M., Leskovec, J., Sosic, R., Hawkins, J., Brownstein, J., Conidi, G., Gunn, J., Gray, J., Zink, A. and Santillana, M. (2018b) Accurate Influenza Monitoring and Forecasting Using Novel Internet Data Streams: A Case Study in the Boston Metropolis. *JMIR Public Health Surveill.* 4(1):e4. DOI: 10.2196/publichealth.8950. [Online]. Available from: https://publichealth.jmir.org/2018/1/e4/pdf. Accessed: 23/04/2018.

Marques-Toledo, C.A., Degener, C.M., Vinhal, L., Coelho, G., Meira, W., Codeço, C.T. and Teixeira, M.M. (2017) Dengue prediction by the web: Tweets are a useful tool for estimating and forecasting Dengue at country and city level. *PLOS Neglected Tropical Diseases*. [Online]. 11(7). Available from:

http://journals.plos.org/plosntds/article?id=10.1371/journal.pntd.0005729. [Accessed: 10/07/2018].

Morin, C.W., Monaghan, A.J., Hayden, M.H., Barrera, R. and Ernst, K. (2015) Meteorologically Driven Simulations of Dengue Epidemics in San Juan, PR. *PLOS Neglected Tropical Diseases.* [Online]. 14;9(8):e0004002. DOI: 10.1371/journal.pntd.0004002. Available from: http://journals.plos.org/plosntds/article?id=10.1371/journal.pntd.0004002 [Accessed: 13/06/2018].

Musoro, J.Z., Zwinderman, A.H., Puhan, M.A., Riet, G.T. and Geskus, R.B. (2016) Validation of prediction models based on lasso regression with multiply imputed data. *BMC Medical Research Methodology*. 14:116. DOI: https://doi.org/10.1186/1471-2288-14-116 Available from: https://bmcmedresmethodol.biomedcentral.com/track/pdf/10.1186/1471-2288-14-116 [Accessed: 23/04/2019].

National Oceanic and Atmospheric Administration (NOAA). (2018a) *Dengue Forecasting*. [Online]. Available from: http://dengueforecasting.noaa.gov/. [Accessed: 08/07/2018].

National Oceanic and Atmospheric Administration (NOAA). (2018a) *Dengue Forecasting Project*. [Online]. Available from: http://dengueforecasting.noaa.gov/docs/project_description.pdf. [Accessed: 08/07/2018].

National Oceanic and Atmospheric Administration. (2018c) *Metadata for Dengue in Iquito, Peru - 2000/2001 to 2008/2009 Seasons*. [Online]. Available from: http://dengueforecasting.noaa.gov/Training/dengue-metadata-iquitos_training.html. [Accessed: 08/07/2018].

National Oceanic and Atmospheric Administration. (2018d) *Metadata for Dengue in San Juan, Puerto Rico - 1990/1991 to 2008/2009 Seasons*. [Online]. Available from: http://dengueforecasting.noaa.gov/Training/dengue-metadata-san_juan_training.html. [Accessed: 08/07/2018].

New Zealand Government. (2018) *Dengue Fever outbreak*. [Online]. Available from: https://www.safetravel.govt.nz/news/solomon-islands-vanuatu-new-caledonia-dengue-fever. [Accessed: 10/07/2018].

Obama White House. (2015) *Back to the Future: Using Historical Dengue Data to Predict the Next Epidemic*. [Online]. Available from: https://obamawhitehouse.archives.gov/blog/2015/06/05/back-future-using-historical-dengue-data-predict-next-epidemic. [Accessed: 08/07/2018].

Python Software Foundation. (2018) *pandas-profiling 1.4.1*. [Online]. Available from: https://pypi.org/project/pandas-profiling/. [Accessed: 09/12/2018].

ResearchGate. (2018) *How to manage MNAR in my study?*. [Online]. Available from: https://www.researchgate.net/post/How_to_manage_MNAR_in_my_study. [Accessed: 10/12/2018].

Sampri, A., Mavragani, A. and Tsagarakis, K.P. (2016) Evaluating Google Trends as a Tool for Integrating the 'Smart Health' Concept in the Smart Cities' Governance in USA. *Procedia Engineering*. 162. p. 585-592.

scikit-learn developers. (2018a) *sklearn.preprocessing.StandardScaler*. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html. [Accessed: 10/12/2018].

scikit-learn developers. (2018b) *sklearn.model_selection.train_test_split*. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html. [Accessed: 10/12/2018].

scikit-learn developers. (2018c) *3.2.4.1.3. sklearn.linear_model.LassoCV*. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LassoCV.html. [Accessed: 10/12/2018].

scikit-learn developers. (2018d) *Model selection: choosing estimators and their parameters*. Available from: https://scikit-learn.org/stable/tutorial/statistical_inference/model_selection.html. [Accessed: 10/12/2018].

scikit-learn developers. (2018e) *Lasso model selection: Cross-Validation / AIC / BIC*. Available from: https://scikit-learn.org/stable/auto_examples/linear_model/plot_lasso_model_selection.html#sphx-glr-auto-examples-linear-model-plot-lasso-model-selection-py. [Accessed: 10/12/2018].

scikit-learn developers. (2018f) *3.3. Model evaluation: quantifying the quality of predictions*. Available from: https://scikit-learn.org/stable/modules/model_evaluation.html. [Accessed: 10/12/2018].

scikit-learn developers. (2018g) *1.13. Feature selection*. Available from: https://scikit-learn.org/stable/modules/feature_selection.html. [Accessed: 10/12/2018].

scikit-learn developers. (2018h) *sklearn.model_selection.TimeSeriesSplit*. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.TimeSeriesSplit.html. [Accessed: 23/04/2019].

Seung-Pyo, J., Hyoung, S.Y. and San, C. (2018) Ten years of research change using Google Trends: From the perspective of big data utilizations and applications. *Technological Forecasting & Social Change*. 130. p. 69–87.

Shi, Y., Liu, X., Kok, S.Y., Rajarethinam, J., Liang, S., Yap, G., Chong, C.S., Lee, K.S., Tan, S.S., Chin, C.K., Lo, A., Kong, W., Ng, L.C. and Cook, A.R. (2016) Three-Month Real-Time Dengue Forecast Models: An Early Warning System for Outbreak Alerts and Policy Decision Support in Singapore. *Environmental Health Perspectives*. 124(9). p. 1369-1375.

Sougata, D., Acebedo, C.M.L. and Chua, M.C.H. (2017) An ensemble prediction approach to weekly Dengue cases forecasting based on climatic and terrain conditions. *Journal of Health and Social Sciences*. 2(3). p. 257-272.

Stack Exchange Inc. (2016a) *Pandas - Alternative to rank() function that gives unique ordinal ranks for a column*. [Online]. Available from: https://stackoverflow.com/questions/39707080/pandas-alternative-to-rank-function-that-gives-unique-ordinal-ranks-for-a-co. [Accessed: 09/12/2018].

Stack Exchange Inc. (2016b) *Fill NaN in pandas column in both direction*. [Online]. Available from: https://stackoverflow.com/questions/34280094/fill-nan-in-pandas-column-in-both-direction. [Accessed: 09/12/2018].

Stack Exchange Inc. (2016c). *Should we normalize before using VarianceThreshold in sklearn?* [Online]. Available from: https://stats.stackexchange.com/questions/253920/should-we-normalize-before-using-variancethreshold-in-sklearn. [Accessed: 09/12/2018].

Stack Exchange Inc. (2016d). *Retain feature names after Scikit Feature Selection*. [Online]. Available from: https://stackoverflow.com/questions/39812885/retain-feature-names-after-scikit-feature-selection. [Accessed: 09/12/2018].

Statistical Programming. (2019) *Predictive Mean Matching Imputation (Theory & Example in R)*. [Online]. Available from: https://statistical-programming.com/predictive-mean-matching-imputation-method/. [Accessed: 23/04/2019].

Strauss, R.A., Castro, J.S., Reintjes, R. and Torres, J.R. (2017) Google dengue trends: An indicator of epidemic behavior. The Venezuelan Case. *International Journal of Medical Informatics*. 104. p. 26–30.

TapVentures. (2017) *Predicting Disease Spread Using Dengue*. [Online]. Available from: http://tapventures-blog.tumblr.com/post/159944890859/predicting-disease-spread-using-dengue. [Accessed: 10/12/2018].

Taylor, J. (2012) *Statistics 202: Data Mining - Linear regression & LASSO*. [Online] Decmber 2012. Available from: http://statweb.stanford.edu/~jtaylo/courses/stats202/restricted/notes/lasso.pdf [Accessed: 23/04/2019].

Teng, Y., Bi, D., Xie, G., Jin, Y., Huang, Y., Lin, B., An, X., Feng, D. and Tong, Y. (2017) Dynamic Forecasting of Zika Epidemics Using Google Trends. *PLOS One*. [Online]. DOI:10.1371/journal.pone.0165085. Available from: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0165085. [Accessed: 11/07/2018].

The International Federation of Red Cross and Red Crescent Societies (2011a). *Peru and Bolivia: Dengue outbreak - DREF operation n° MDR46001 18 February 2011*. [Online]. Available from: https://reliefweb.int/sites/reliefweb.int/files/resources/1507AAFA08C7F39E8525783B00764030-Full_Report.pdf. [Accessed: 05/12/2018].

The International Federation of Red Cross and Red Crescent Societies (2011b). *Peru and Bolivia: Dengue outbreak - DREF operation n° MDR46001 Update n° 1 13 May 2011*. [Online]. Available from: http://www.ifrc.org/docs/appeals/11/MDR46001du1.pdf. [Accessed: 05/12/2018].

The International Federation of Red Cross and Red Crescent Societies (2011c). *Peru and Bolivia: Dengue outbreak - DREF operation n° MDR46001 21 September 2011*. [Online]. Available from: https://reliefweb.int/sites/reliefweb.int/files/resources/Full_Report_2391.pdf. [Accessed: 05/12/2018].

The pandas project. (2018a) *Working with missing data*. [Online]. Available from: http://pandas.pydata.org/pandas-docs/stable/missing_data.html#interpolation. [Accessed: 08/12/2018].

The pandas project. (2018b) *pandas.DataFrame.corr*. [Online]. Available from: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.corr.html. [Accessed: 08/12/2018].

The pandas project. (2018c) *pandas.DataFrame.describe*. [Online]. Available from: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.describe.html. [Accessed: 08/12/2018].

The Pennsylvania State University. (2018) *5.4 - The Lasso*. [Online]. Available from: https://onlinecourses.science.psu.edu/stat857/node/158/. [Accessed: 08/12/2018].

The Scipy community (2015) scipy.stats.rankdata. [Online]. Available from: https://docs.scipy.org/doc/scipy-0.16.0/reference/generated/scipy.stats.rankdata.html. [Accessed: 09/12/2018].

The SciPy community. (2018) *Interpolation (scipy.interpolate)*. [Online]. Available from: https://docs.scipy.org/doc/scipy/reference/tutorial/interpolate.html. [Accessed: 08/12/2018].

Tibshirani, R. (2011) 'Regression shrinkage and selection via the lasso: a retrospective' in Dunson, D. and Wood, S. (eds.) *Journal of Royal Statistics Society: Statistical Methodology Series B*. New Jersey: John Wiley & Sons. p. 273–282.

Tibshirani, R.J. (2012) *The Lasso Problem and Uniqueness*. [Online]. November 2012. Available from: https://arxiv.org/pdf/1206.0313.pdf. [Accessed: 23/04/2019].

Torres, J.R., Orduna, T.A., Piña-Pozas, M., Vázquez-Vega, D. and Sarti, E. (2017) Epidemiological Characteristics of Dengue Disease in Latin America and in the Caribbean: A Systematic Review of the Literature. *Journal of Tropical Medicine*. 2017. DOI: https://doi.org/10.1155/2017/8045435.

United States Census Bureau. (2019) *QuickFacts - San Juan Municipio, Puerto Rico*. [Online]. Available from: https://www.census.gov/quickfacts/fact/table/sanjuanmunicipiopuertorico/PST045218 [Accessed: 23/04/2019].

Venkatramanan, S., Lewis, B., Chen, J., Higdon, D., Vullikanti, A. and Marathe, M. (2018) Using data-driven agent-based models for forecasting emerging infectious diseases. *Epidemics*. 22. p. 43–49.

Waskom, M. (2018a). *seaborn.violinplot*. [Online]. Available from: https://seaborn.pydata.org/generated/seaborn.violinplot.html. [Accessed: 09/11/2018].

Waskom, M. (2018b) *seaborn.heatmap*. [Online]. Available from: https://seaborn.pydata.org/generated/seaborn.heatmap.html. [Accessed: 09/11/2018].

World Atlas. (2018) *What Languages Are Spoken In Puerto Rico?*. [Online]. Available from: https://www.worldatlas.com/articles/what-languages-are-spoken-in-puerto-rico.html. [Accessed: 11/07/2018].

World Health Organization (2018) *Dengue and severe dengue*. [Online]. Available from: https://www.who.int/news-room/fact-sheets/detail/dengue-and-severe-dengue. [Accessed: 12/12/2018].

World Meteorological Organization (2017). *Nowcasting*. [Online]. Available from: http://www.wmo.int/pages/prog/amp/pwsp/Nowcasting.htm. [Accessed: 09/11/2018].

Yamana, T.K., Kandula, S. and Shaman, J. (2016) Superensemble forecasts of dengue outbreaks. *Journal of The Royal Society Interface*. 13(123). DOI: 10.1098/rsif.2016.0410.

Yang, S., Kou, S.C., Lu, F., Brownstein, J.S., Brooke, N. and Santillana, M. (2017) Advances in using Internet searches to track dengue. *PLOS Computational Biology*. [Online]. Available from: http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005607. [Accessed: 09/07/2018].

Yilmaz, N. (2017) *Handling Missing Data*. [Online]. Available from: https://medium.com/@numanyilmaz61/handling-missing-data-93d3ce5d0161. [Accessed: 09/12/2018].

Zou, B., Lampos, V. and Cox, I. (2018) *Multi-Task Learning Improves Disease Models from Web Search*. In Proceedings of the 2018 World Wide Web Conference (WWW '18). Lyon, France: ACM. p. 87-96.

APPENDICES

APPENDIX 1 – Google Trends API

Access to Google Health Trends API was applied via the "Google Trends API Request form" (Google, 2019) and further communicated with trends-api-support@google.com on the access key, etc.

Project name: My Project 62801
Project ID: sonic-dialect-219408
Project number: 115790834084

Instructions and attached Jupyter notebook template:
https://sites.google.com/a/google.com/health-trends-api-getting-started-guide/?pli=1

Related files in CD:

- Google Trends API - Attached Jupyter notebook Sample.py
- Google Trends API - Dengue.py
- peru_region_loreto - Comparison multi window.xlsx

The reply from trends-api-support trends-api-support@google.com dated
28 Dec 2018, 00:04 indicated that

"We don't support city level data - It's either country level, region level (i.e states in the US), or dma level (metropolitan areas) which is only available for the US.

You can try playing with the external Google Trends product to find the answer to this question, or you may refer to https://en.wikipedia.org/wiki/ISO_3166-2#Current_codes for the supported values."

Therefore, the city level data of San Juan and Iquitos cannot be extracted.

APPENDIX 2 – Feature Selection Process for sj_train

The original table can be found in the CD attached > Features Selection.xlsx > worksheet "sj_train".

	Distribution		Missing Values	High Correlation / Similar Boxplot			Decision 1 - Based on EDA		Decision 2 - Based on Low variance threshold	Significant lags vs. LnAdvance4W_normal_total_cases (weeks)			Reason to drop
	Skewness	Histogram	Count / Percentage	Panda Profilling	Pearson Correlation heatmap	Boxplot	Drop ?	Transformation	Drop?	Positive CCF	Negative CCF	Oscillation?	
Total missing			1.5%										
Variables													
year	(0.00)						Yes						Not useful
weekofyear	(0.01)						No	Standardization	No				
ndvi_ne	(0.02)		191 / 20.5%			ndvi_ne vs. ndvi_nw	Yes						1) Collinear with ndvi_ne 2) High missing values percentage
ndvi_nw	(0.09)		48 / 5.2%			ndvi_ne vs. ndvi_nw	No	Standardization	No	0-32		No	
ndvi_se	0.21		19 / 2.0%		ndvi_se vs. ndvi_sw	ndvi_se vs. ndvi_sw	No	Standardization	No		3, 5-21	No	
ndvi_sw	0.15		19 / 2.0%		ndvi_se vs. ndvi_sw	ndvi_se vs. ndvi_sw	Yes						Collinear with ndvi_se
precipitation_amt_mm	2.61	Right-skewed	9 / 1.0%			precipitation_amt_mm vs. reanalysis_sat_precip_amt_mm	No	ln(1+x) Standardization	No		19-32	Yes	
reanalysis_precip_amt_kg_per	5.57	Right-skewe	6 / 0.6%				No	ln(1+x) Standardization	No	0-13	18-32	Yes	
reanalysis_relative_humidity_percent	(0.20)		6 / 0.6%				No	Standardization	No	0-17	23-32	Yes	
reanalysis_sat_precip_amt_mm	2.61	Right-skewed		highly correlated with precipitation_amt_mm (ρ = 1)		precipitation_amt_mm vs. reanalysis_sat_precip_amt_mm	Yes						Collinear with precipitation_amt_mm
reanalysis_specific_humidity_g_per_kg	(0.48)		6 / 0.6%		reanalysis_specific_humidity_g_per_kg vs. station and reanalysis temperatures except tdtk (diural)		No	Standardization	No	0-12	19-32	Yes	
station_avg_temp_c	(0.32)		6 / 0.6%		station vs. reanalysis temperatures except tdtk (diural)		No	Standardization	No	0-19	22-32	Yes	
station_diur_temp_rng_c	0.10		6 / 0.6%				No	Standardization	No	10-32		No	
station_max_temp_c	(0.45)		6 / 0.6%		station vs. reanalysis temperatures except tdtk (diural)		No	Standardization	No	0-21	25-32	Yes	
station_min_temp_c	(0.39)		6 / 0.6%		station vs. reanalysis temperatures except tdtk (diural)		No	Standardization	No	0-18	21-32	Yes	
station_precip_mm	2.63	Right-skewed	6 / 0.6%		station vs. reanalysis temperatures except tdtk (diural)		No	ln(1+x) Standardization	No	0-7	14-32	Yes	
normal_reanalysis_dew_point_temp_C	(0.64)			highly correlated with reanalysis_specific_humidity_g_per_kg (ρ = 0.99853)	station vs. reanalysis temperatures except tdtk (diural)		Yes						Collinear with reanalysis_specific_humidity_g_per_kg
normal_reanalysis_max_air_temp_C	(0.17)		6 / 0.6%		station vs. reanalysis temperatures except tdtk (diural)		No	Standardization	No	0-16	19-32	Yes	
normal_reanalysis_min_air_temp_C	(0.55)		6 / 0.6%		station vs. reanalysis temperatures except tdtk (diural)		No	Standardization	No	0-15	19-32	Yes	
normal_reanalysis_avg_temp_C	(0.23)			highly correlated with normal_reanalysis_min_air_temp_C (ρ = 0.93911)	station vs. reanalysis temperatures except tdtk (diural)	normal_reanalysis_avg_temp_C vs normal_reanalysis_air_temp_C	Yes						Collinear with normal_reanalysis_min_air_temp_C
normal_reanalysis_tdtr_C	0.68		6 / 0.6%		station vs. reanalysis temperatures except tdtk (diural)		No	Standardization	No	0-8, 30-32		No	
normal_reanalysis_air_temp_C	(0.23)			highly correlated with normal_reanalysis_avg_temp_C (ρ = 0.99749)	station vs. reanalysis temperatures except tdtk (diural)	normal_reanalysis_avg_temp_C vs normal_reanalysis_air_temp_C	Yes						Collinear with normal_reanalysis_avg_temp_C
normal_total_cases	4.51	Right-skewed			normal_total_case vs. Advance4W_normal_total_case		Yes						Collinear with Advance4W_normal_total_cases
Advance4W_normal_total_cases	4.51	Right-skewed			normal_total_case vs. Advance4W_normal_total_case		No	ln(1+x) Standardization	No				

APPENDIX 3 – Feature Selection Process for iq_train

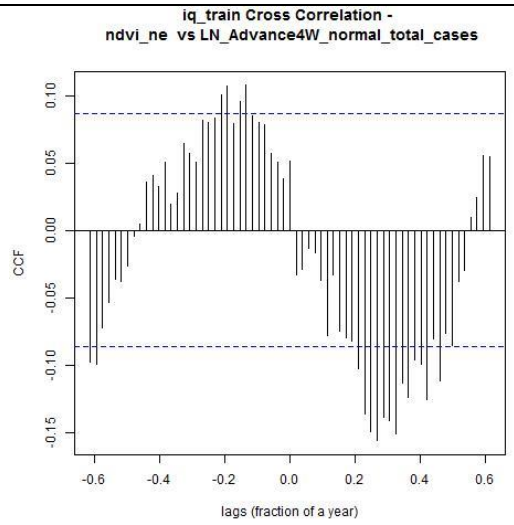
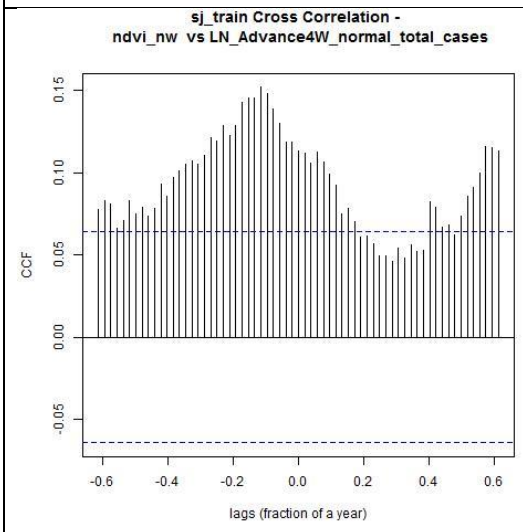
The original table can be found in the CD attached > Features Selection.xlsx > worksheet "iq_train".

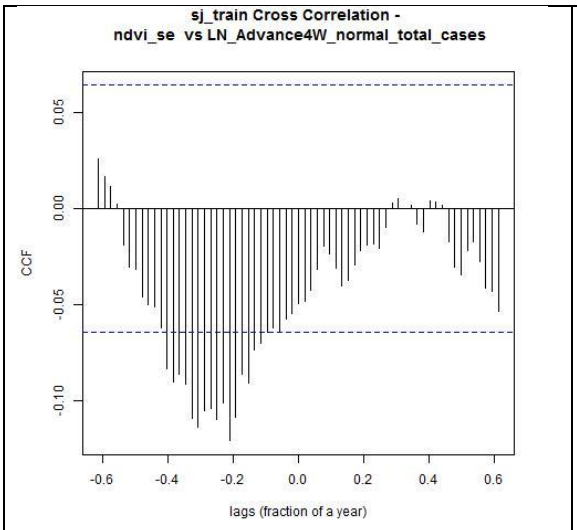
	Distribution		Missing Value	High Correlation / Similar Boxplot			Decision 1 - Based on EDA		Decision 2 - Based on Low variance threshold	Significant lags vs. LnAdvance4W_normal_total_cases (weeks)			Reason to drop
	Skewness	Histogram	Count / Percentage	Panda Profilling	Pearson Correlation heatmap	Boxplot	Drop?	Transformation	Drop?	Positive CCF	Negative CCF	Oscillation?	
Total missing			1.2%										
Variables													
year	0.00						Yes						
weekofyear	(0.00)						No	Standardization	No				Not useful
ndvi_ne	0.23		3 / 0.6%		4 sets of ndvi	ndvi_ne vs. ndvi_nw	No	Standardization	No	7-8, 10-11	31-32	Yes	
ndvi_nw	0.23		3 / 0.6%		4 sets of ndvi	ndvi_ne vs. ndvi_nw	Yes						Collinear with ndvi_ne
ndvi_se	0.27		3 / 0.6%		4 sets of ndvi	ndvi_se vs. ndvi_sw	Yes						Collinear with ndvi_sw
ndvi_sw	0.27		3 / 0.6%		4 sets of ndvi	ndvi_se vs. ndvi_sw	No	Standardization	No	4-18, 19-21	31-32	Yes	
precipitation_amt_mm	0.60				reanalysis_sat_precip_amt_mm vs. precipitation_amt_mm	precipitation_amt_mm vs. reanalysis_sat_precip_amt_mm	No	Standardization	No	0-2	9-13, 15-23	Yes	
reanalysis_precip_amt_kg_per	2.01	Right-skewed	4 / 0.8%				No	ln(1+x) Standardization	No	0-5, 31	13, 17-19	Yes	
reanalysis_relative_humidity_percent	(1.10)		4 / 0.8%				No	Standardization	No	0-2, 27-32	13	Yes	
reanalysis_sat_precip_amt_mm	0.60			highly correlated with precipitation_amt_mm (ρ = 1)	reanalysis_sat_precip_amt_mm vs. precipitation_amt_mm	precipitation_amt_mm vs. reanalysis_sat_precip_amt_mm	Yes						Collinear with precipitation_amt_mm
reanalysis_specific_humidity_g_per_kg	(0.66)		4 / 0.8%		reanalysis_specific_humidity_g_per_kg vs. normal_reanalysis_dew_point_temp_C		No	Standardization	No	0-8	17-23	Yes	
station_avg_temp_c	(0.91)		37 / 7.2%				No	Standardization	No	0-9	15, 18 - 32	Yes	
station_diur_temp_rng_c	(0.07)						No	Standardization	No		24-32	No	
station_max_temp_c	0.63		14 / 2.7%				No	Standardization	No	0-7	22-32	Yes	
station_min_temp_c	(1.15)		8 / 1.6%				No	Standardization	No	0-5	12-27	Yes	
station_precip_mm	2.21	Right-skewed	16 / 3.1%				No	ln(1+x) Standardization	No		10-13, 18	No	
normal_reanalysis_dew_point_temp_C	(0.88)			highly correlated with reanalysis_specific_humidity_g_per_kg (ρ = 0.99778)	reanalysis_specific_humidity_g_per_kg vs. normal_reanalysis_dew_point_temp_C		Yes			0-8	18-21, 23	Yes	Collinear with reanalysis_specific_humidity_g_per_kg
normal_reanalysis_max_air_temp_C	0.16		4 / 0.8%				No	Standardization	No	6-14	25-32	Yes	
normal_reanalysis_min_air_temp_C	(0.96)		4 / 0.8%				No	Standardization	No	0-7	13, 15, 17-28	Yes	
normal_reanalysis_avg_temp_C	(0.11)		4 / 0.8%		normal_reanalysis_air_temp_C vs. normal_reanalysis_avg_temp_C		No	Standardization	No	0-11	23-32	Yes	
normal_reanalysis_tdtr_C	0.28		4 / 0.8%				No	Standardization	No	10-20	0-1, 27-32	Yes	
normal_reanalysis_air_temp_C	0.10			highly correlated with normal_reanalysis_avg_temp_C (ρ = 0.97332)	normal_reanalysis_air_temp_C vs. normal_reanalysis_avg_temp_C		Yes			0-12	23-32	Yes	Collinear with normal_reanalysis_avg_temp_C
normal_total_cases	4.06	Right-skewed					Yes						Collinear with Advance4W_normal_total_cases
Advance4W_normal_total_cases	4.06	Right-skewed					No	ln(1+x) Standardization	No				

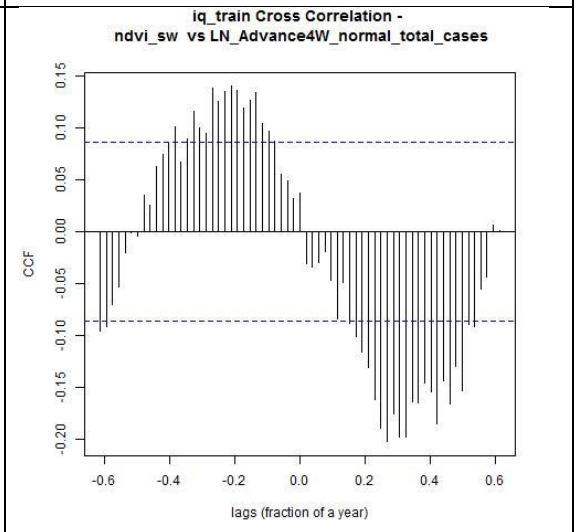
APPENDIX 4 - Cross Correlation Functions from R

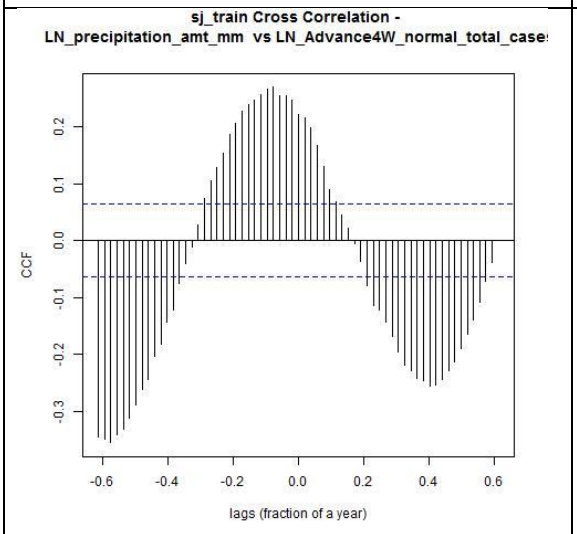
The following are the output from es

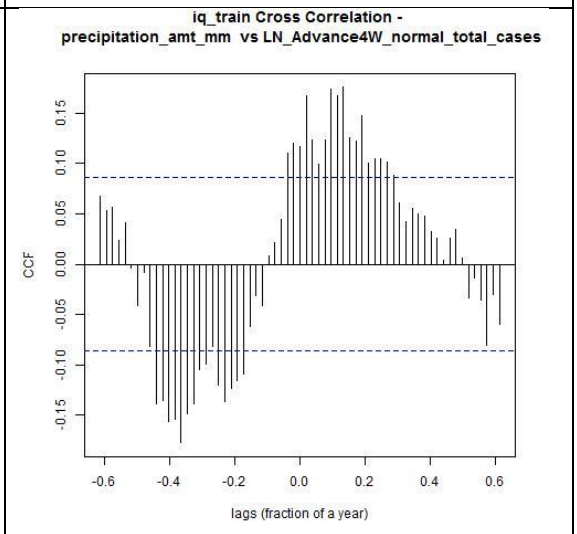
- sj_train - Cross Correlation Function to Determine Significant Lags v2.R
- iq_train - Cross Correlation Function to Determine Significant Lags v2.R

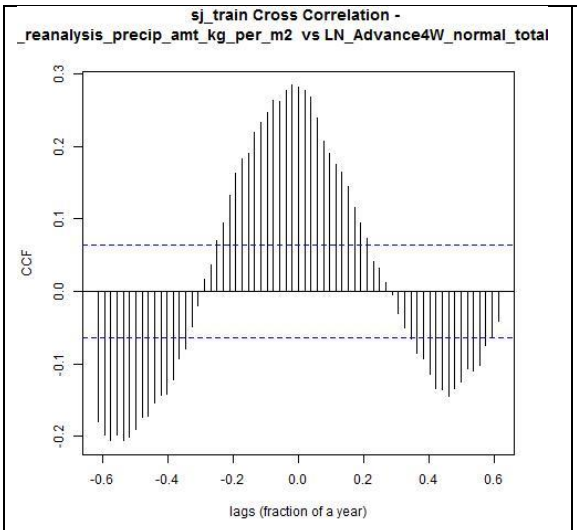
sj_train	iq_train
	
	

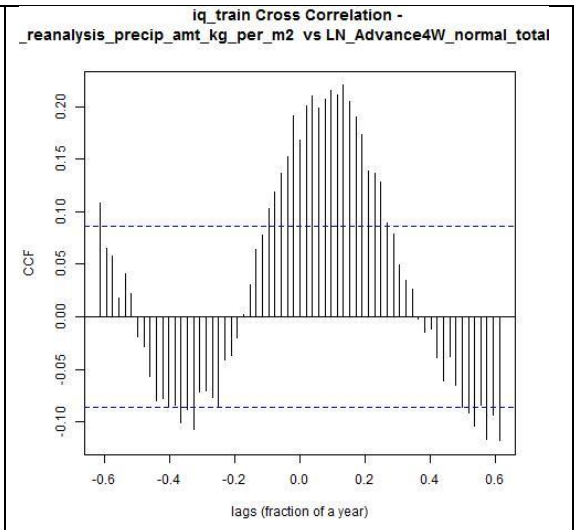

sj_train Cross Correlation -
ndvi_se vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)

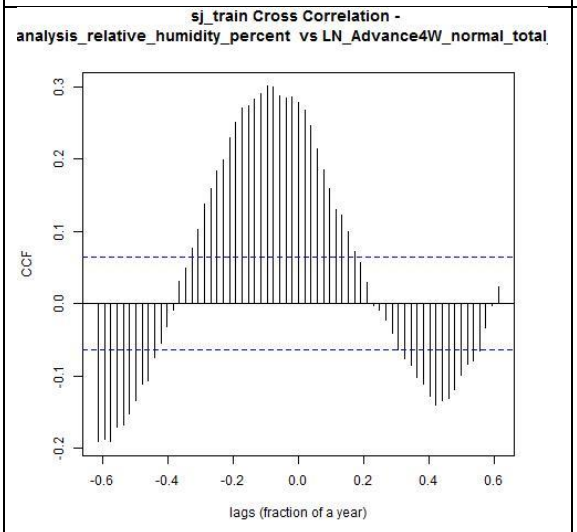

iq_train Cross Correlation -
ndvi_sw vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)

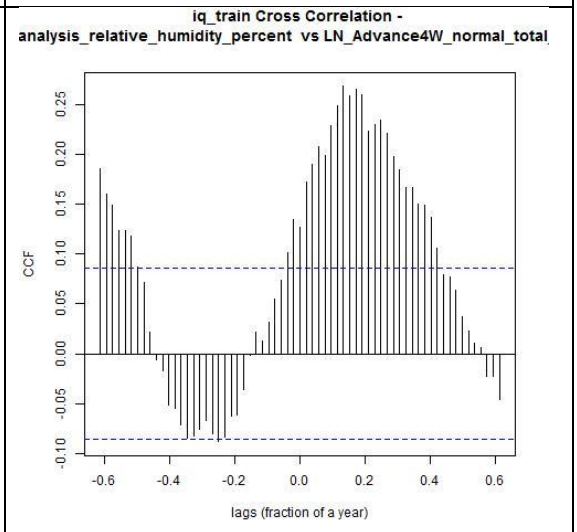

sj_train Cross Correlation -
LN_precipitation_amt_mm vs LN_Advance4W_normal_total_case
CCF
lags (fraction of a year)

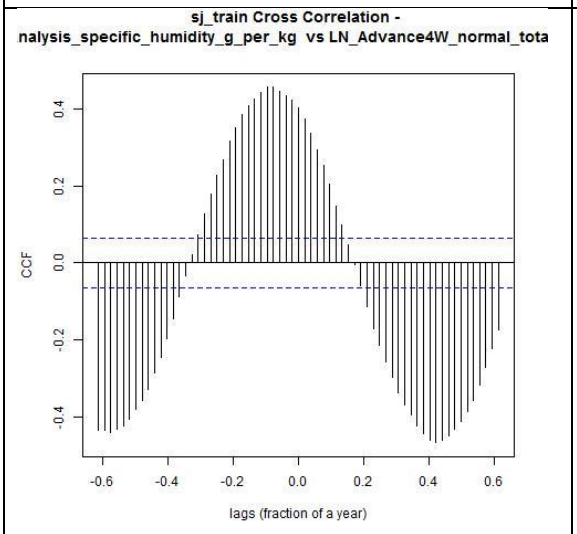

iq_train Cross Correlation -
precipitation_amt_mm vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)

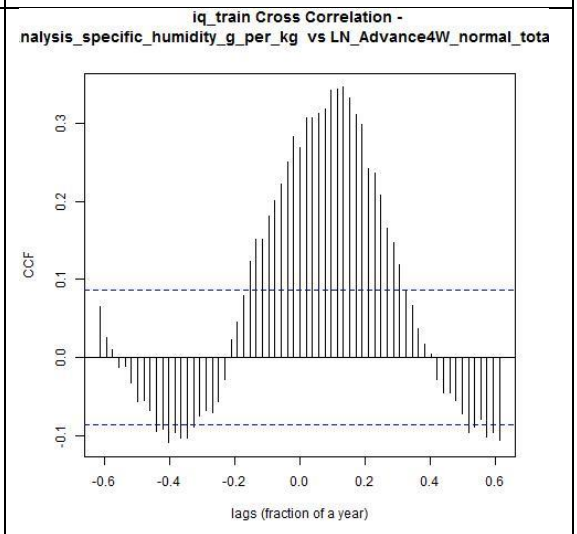

sj_train Cross Correlation -
_reanalysis_precip_amt_kg_per_m2 vs LN_Advance4W_normal_total
CCF
lags (fraction of a year)

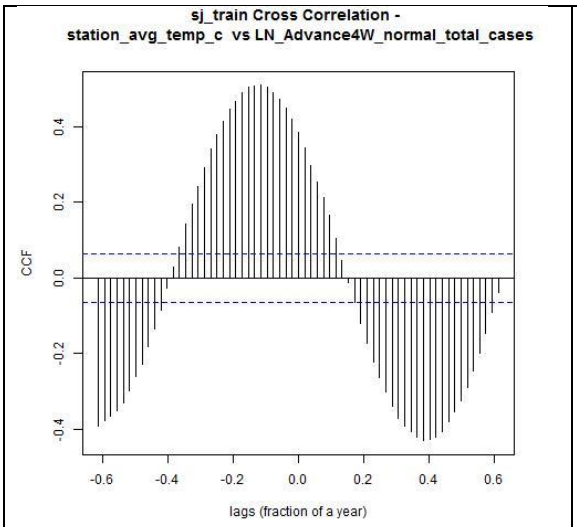

iq_train Cross Correlation -
_reanalysis_precip_amt_kg_per_m2 vs LN_Advance4W_normal_total
CCF
lags (fraction of a year)

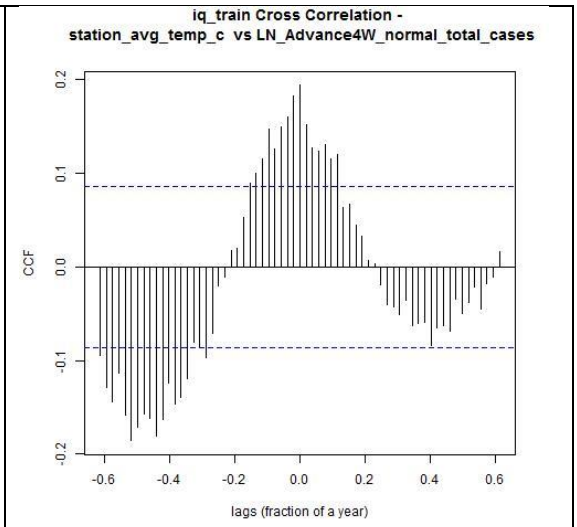

sj_train Cross Correlation -
analysis_relative_humidity_percent vs LN_Advance4W_normal_total
CCF
lags (fraction of a year)

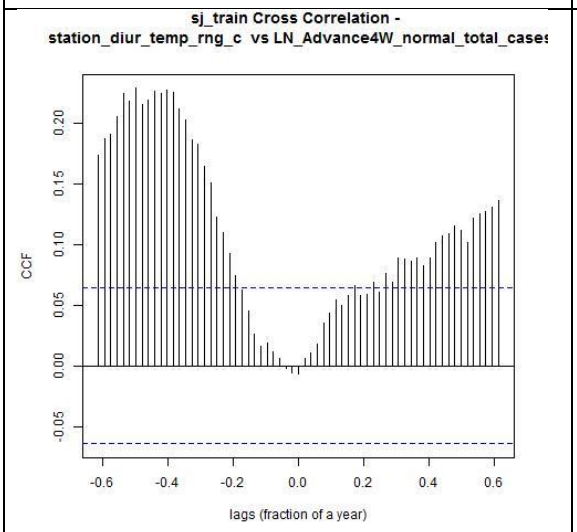

iq_train Cross Correlation -
analysis_relative_humidity_percent vs LN_Advance4W_normal_total
CCF
lags (fraction of a year)

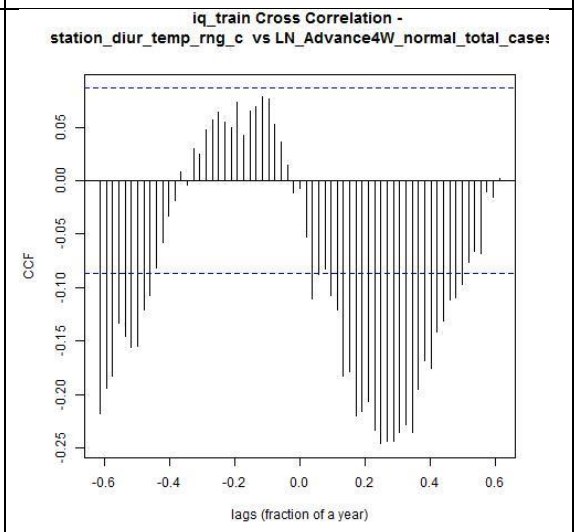

sj_train Cross Correlation -
nalysis_specific_humidity_g_per_kg vs LN_Advance4W_normal_tota
CCF
lags (fraction of a year)

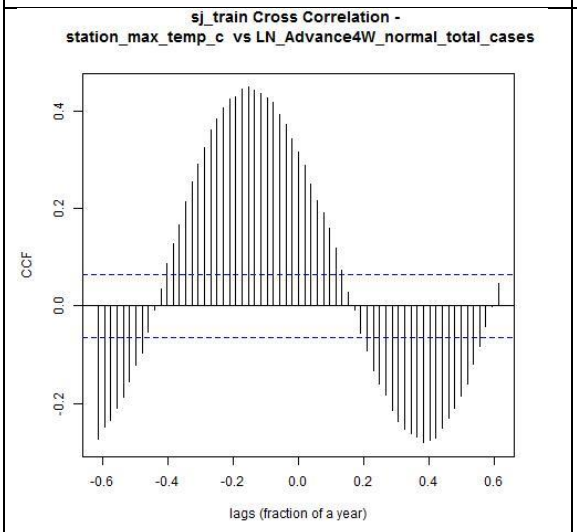

iq_train Cross Correlation -
nalysis_specific_humidity_g_per_kg vs LN_Advance4W_normal_tota
CCF
lags (fraction of a year)

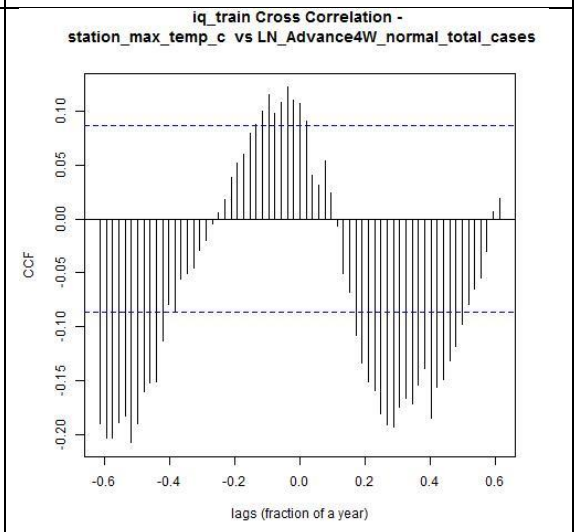

sj_train Cross Correlation -
station_avg_temp_c vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)

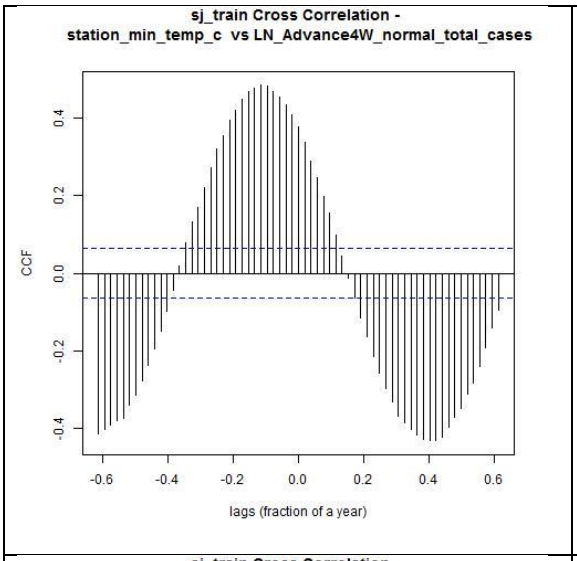

iq_train Cross Correlation -
station_avg_temp_c vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)

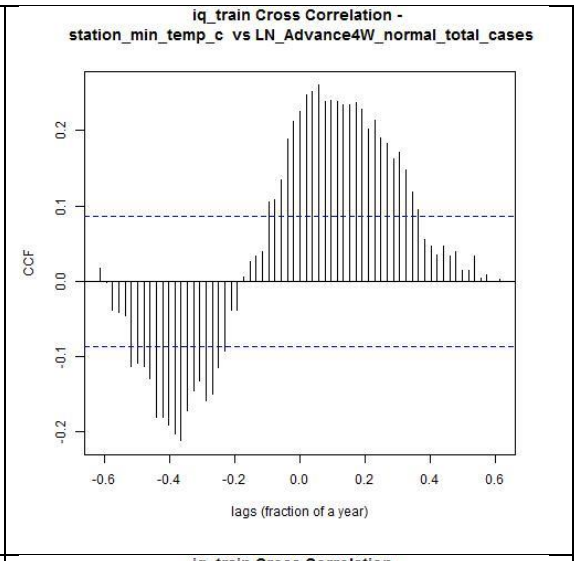

sj_train Cross Correlation -
station_diur_temp_rng_c vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)

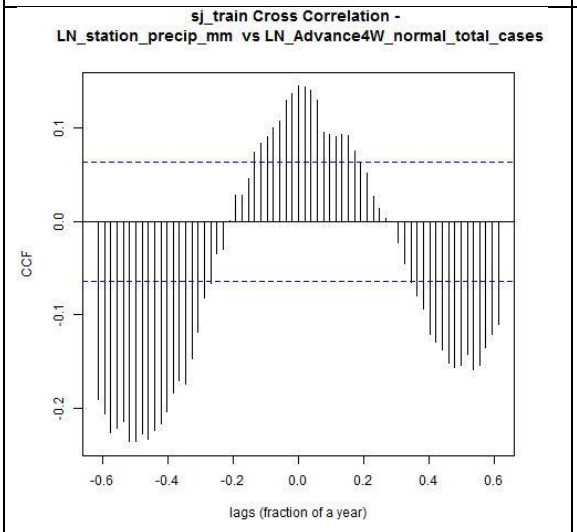

iq_train Cross Correlation -
station_diur_temp_rng_c vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)

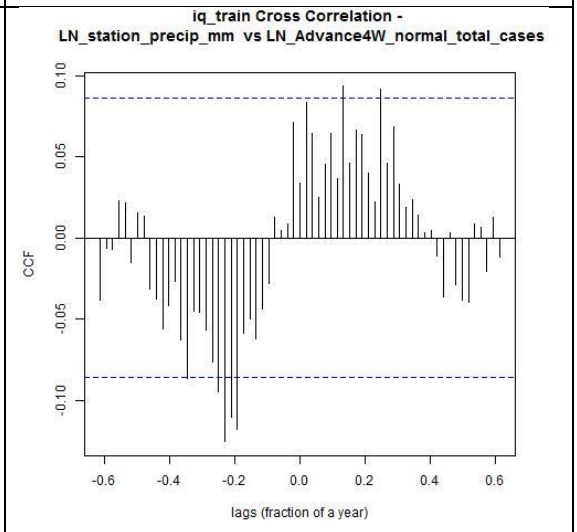

sj_train Cross Correlation -
station_max_temp_c vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)

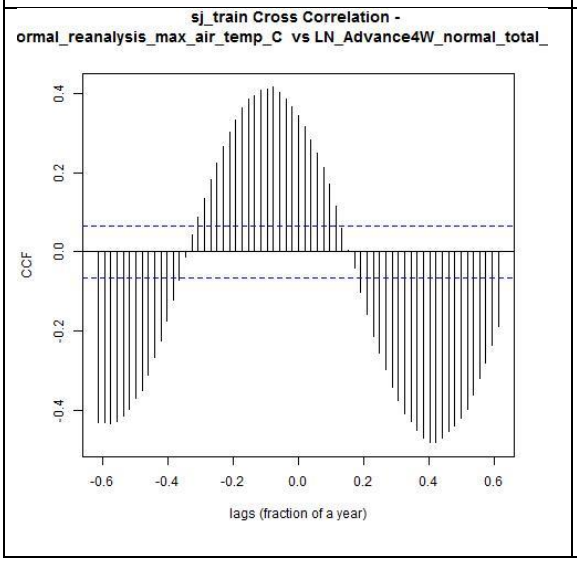

iq_train Cross Correlation -
station_max_temp_c vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)

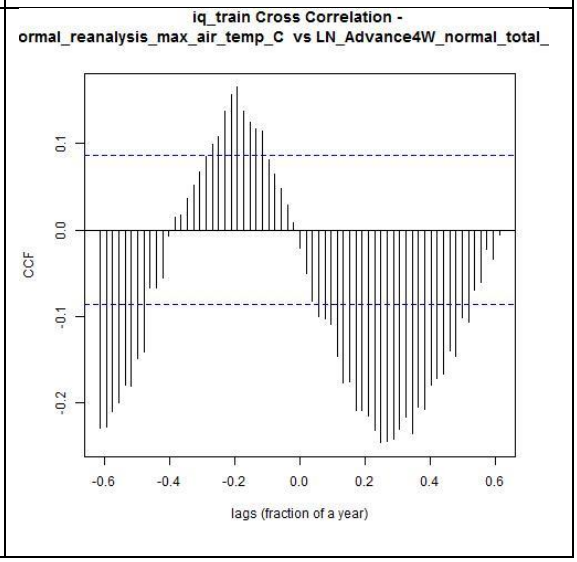

sj_train Cross Correlation -
station_min_temp_c vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)

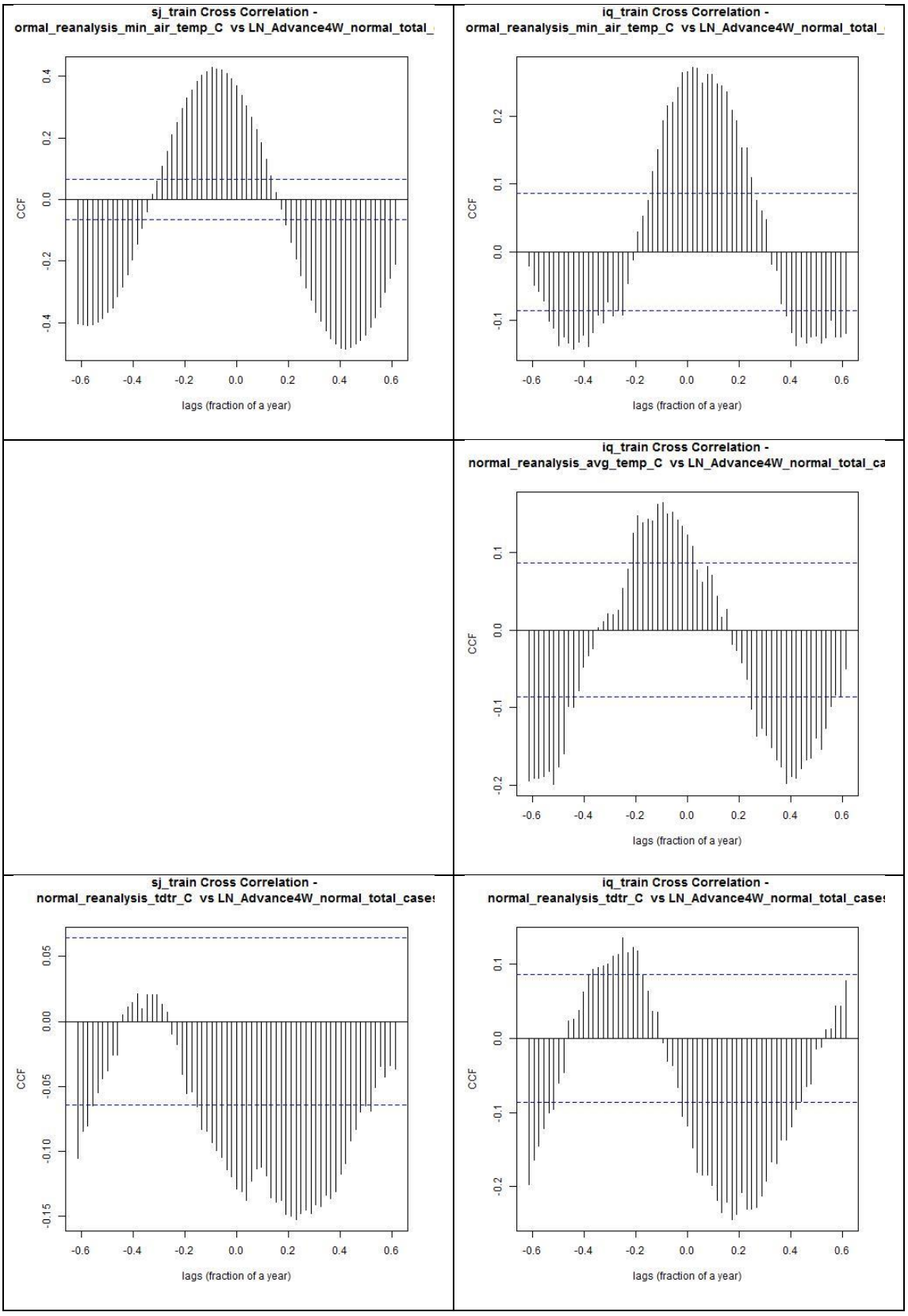

iq_train Cross Correlation -
station_min_temp_c vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)


sj_train Cross Correlation -
LN_station_precip_mm vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)


iq_train Cross Correlation -
LN_station_precip_mm vs LN_Advance4W_normal_total_cases
CCF
lags (fraction of a year)


sj_train Cross Correlation -
ormal_reanalysis_max_air_temp_C vs LN_Advance4W_normal_total_
CCF
lags (fraction of a year)


iq_train Cross Correlation -
ormal_reanalysis_max_air_temp_C vs LN_Advance4W_normal_total_
CCF
lags (fraction of a year)


sj_train Cross Correlation -
ormal_reanalysis_min_air_temp_C vs LN_Advance4W_normal_total_
CCF
lags (fraction of a year)
iq_train Cross Correlation -
ormal_reanalysis_min_air_temp_C vs LN_Advance4W_normal_total_
CCF
lags (fraction of a year)
iq_train Cross Correlation -
normal_reanalysis_avg_temp_C vs LN_Advance4W_normal_total_ca
CCF
lags (fraction of a year)
sj_train Cross Correlation -
normal_reanalysis_tdtr_C vs LN_Advance4W_normal_total_case
CCF
lags (fraction of a year)
iq_train Cross Correlation -
normal_reanalysis_tdtr_C vs LN_Advance4W_normal_total_case
CCF
lags (fraction of a year)

APPENDIX 5 – Research Paper

Nowcasting using Least Absolute Shrinkage Selector Operator (LASSO) to Predict Dengue Occurrence in San Juan and Iquitos as Part of Disease Surveillance System

Tang Sui Lan
Faculty of Computing, Engineering and Technology,
Asia Pacific University of Technology & Innovation.

ABSTRACT

Dengue which was first detected mainly in South East Asia during 1940s is now a serious public health concern across the subtropical and temperate regions of Americas, Europe and China due to the change in global climate and international travel. This research aims to forecast dengue cases four weeks in advance for San Juan, Puerto Rico and Iquitos, Peru using Least Absolute Shrinkage Selector Operator (LASSO) model. A literature review of dengue prediction, nowcasting and Google Trends was conducted. Two models were built for each city to evaluate the value added from feature engineering of climatic variables. Time series model diagnostic plots were used to guide model selection between the two set of models. Significant climatic variables that could impact the dengue cases in each city were identified accordingly. Discussion on the limitations of Google Trends data and MICE imputation; time series cross validation methodology, along with the challenges of over-shrinkage and non-unique solutions of Lasso were presented as well. LASSO's flexibility in incorporating a variety of predictors and its ease of interpretation present LASSO as a compelling case against the general predictive models. Public health regulators could make use of such nowcasting model to facilitate the timing of vector control and public health campaigns along with the allocation of medical resource to cope with potential dengue outbreaks.

Keywords:
Dengue, LASSO, Nowcasting, Iquitos, San Juan, Google Trends.

1. Introduction

The dengue disease remains deadly at 2.5% fatality rate as potent vaccine for the different dengue virus serotypes has yet to be developed. This research aims to forecast dengue based on nowcasting technique and Google Trends to improve real time surveillance system. Least Absolute Shrinkage Selector Operator (LASSO), a form of penalized regression was used to forecast the dengue cases of two South American cities, four weeks in advance.

Section 2 covered the literature review of Lasso, nowcasting, Google Trends and the related work on the two cities. Section 3 detailed the research methodology, in terms of integration of multiple data source of different reporting frequency, explorative data analysis, feature selection and feature engineering. The results of the 4 Lasso models were accounted in Section 4. Section 5 discussed Google Trends data limitation, time series validation, Lasso models comparison, nowcasting benchmarking and future recommendations.

2. Literature Review

2.1 Least Absolute Selection and Shrinkage Operator (LASSO)

Lasso $\ell 1$ uses the $\|\beta\|_1$ penalty while Ridge $\ell 2$ uses the $\|\beta\|_2^2$ penalty (Kane, 2015) in handling multi-collinearity among the predictors.

$$\hat{\beta}^{\text{lasso}} = \operatorname{argmin} \|y - X\beta|_2^2 + \lambda \|\beta\|_1$$
$$\hat{\beta}^{\text{ridge}} = \operatorname{argmin} \|y - X\beta\|_2^2 + \lambda \|\beta\|_2^2 \quad (1)$$

However, Ridge does not yield parsimonious models as coefficients are reduced but not zerolized (left hand side of Figure 2.1). Lasso overcomes Ridge's weakness in feature selection by forcing the least important coefficients to zero. The most influential predictor will be identified by Lasso as its coefficient will need the largest λ to be zerolized.

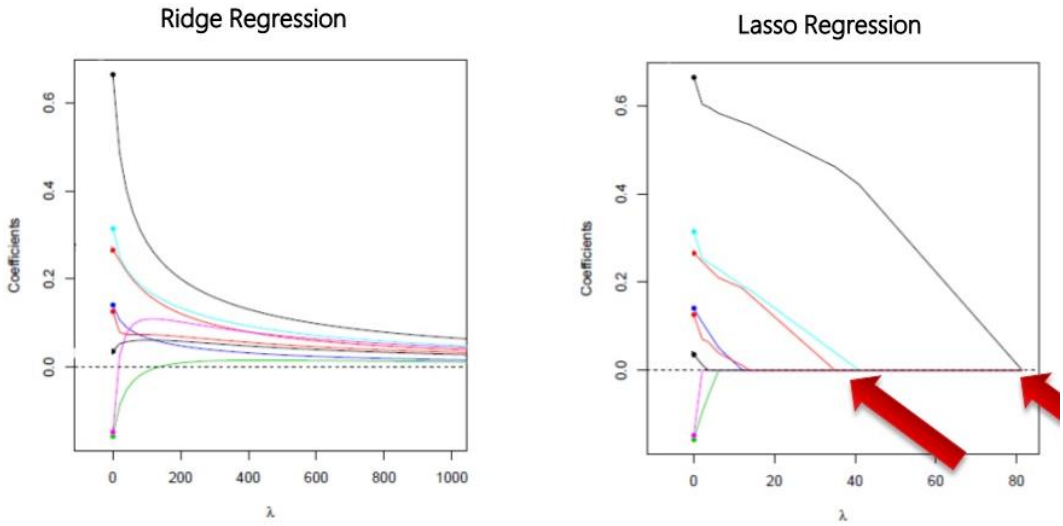


Figure 2.1: Ridge Trace for Ridge Regression vs. Lasso Regression (Kane, 2015).

Lasso models could have lower Mean Square Error (MSE) in comparison to Ridge as influential predictors which are correlated with the chosen predictors are zerolized by Lasso. Elastic Net, a new Ridge-Lasso hybrid tries to balance between the 2 models by allowing grouped selection of correlated features (Kane, 2015; Jain, 2017). Lasso characterizes a convex problem (Tibshirani, 2011) and has higher efficiency when it comes to storage and computation (Jain, 2017).

2.3 Nowcasting with LASSO

Nowcasting is the methodology of using currently available predictors' data to extrapolate occurrence of an interested outcome in near future. Nowcasting which was originated in meteorology (World Meteorological Organization, 2017) is now being adopted in other fields such as finance (European Central Bank, 2013; CRAN, 2018) and medicine (Musuro et al., 2014).

The LASSO-based dengue nowcasting framework operated by Singapore's National Environment Agency (NEA) was documented by Shi et al. (2016). Multiple data streams were integrated on weekly basis – dengue cases, Breeding Percentage (BP, the dominance of Ae. Aegypti among the breeding sites), weather (temperature and humidity), population from midyear census, seasonality and trend decomposition by Breaks For Additive Seasonal and Trend (BFAST) algorithm in R. A total of 226 covariates (with lags up to 20 weeks) were

used to construct 12 LASSO sub-models corresponding to forecast window of 1-12 weeks ahead in a comprehensive training-validation-testing cross validation. A final ensemble model was built to derive an overall prediction for next 12 weeks.

Chen et al. (2018) evaluated LASSO algorithm in nowcasting infectious diseases in Japan (chickenpox and HFMD), Singapore (dengue and HFMD), Taiwan (dengue) and Thailand (chickenpox, dengue and malaria). Wavelet analyses was first performed to study the periodicity (amplitude, time and frequency) of each disease in each locality, out of which lags of disease incidence and climatic variables up to 26 weeks were determined as autoregressive predictors. The dengue incidences underwent logarithmic transformation while the predictors where standardized prior to modelling. Multiple LASSO models were created for each disease in different countries for 1-26 weeks ahead. The optimal constraint parameters were determined using ten-fold cross validation in R. The ability of models to act as early warning system was tested by identifying the threshold corresponding to previous year's 75^{th} percentile high incidence, alongside MAPE, RMSE and R-squared.

2.4 Google Trends (GTs)

GTs sparked great interest as a real time explanatory variable in epidemiology research on the premise that the more a person is affected by a disease, the more likely that he or she will query Google for related information (Yang et al, 2017). Google hosted the Google Flu Trends (GFT) and Google Dengue Trends (GDT) for limited number of countries from 2008 until 2014 (Google, 2018).

An influenza focused ARGO (AutoRegressive model with GOogle search queries as exogenous variables) model was extended to predict national level dengue cases in Brazil, Mexico, Singapore, Taiwan and Thailand (Yang et al, 2017). Autogressive models were built for each country using lags of 1-12 and 24 months under a rolling two year window. Next, top ten dengue-related search terms for respectively country were identified and incorporated into the earlier autoregressive models to form ARGO. Lasso was tweaked to prevent the coefficient of important time lags from being zerolized. The authors attributed the outlying Taiwan model performance to the country having minimal dengue occurrence until the 2014-2015 outbreak plus Google is not the leading search engine in Taiwan. Further tests revealed that ARGO is robust to both the sampling variation of GTs (by collecting the search terms in

ten different sessions) and the scenario that reporting delay caused the latest month dengue cases to be unavailable as autoregressive term.

Teng et al. (2017) appraised Google Trends (GTs) as a source of real time, external regressor to complement time series model in predicting global Zika virus disease (ZVD). The augmented ARIMA model registered lower Akaike Information Criterion (AIC) than the linear baseline model and the basic ARIMA model.

2.5 Related work on San Juan and Iquitos from the competition

2.5.1 Non-ensemble models

Laureano-Rosario et al. (2018) made use of the climatic and population variables from NOOA to forecast the weekly dengue incident rate (per 100,000) for population at risk (those younger than 24 years old) and vulnerable population (small children and old folks). Sea Surface Temperature (SST) data were extracted from NOAA Advanced Very High Resolution Radiometer (AVHRR) as an additional predictor. The ANN from Radar Pluvial flooding Identification for Drainage System (RAPIDS) was modified with factor, a = 3 and genetic algorithm II (NSGA-II) to create two ANN models for the two population subgroups. The ANNs were run as binary models on whether there would be a potential dengue outbreak against predetermined thresholds for the each population subgroups. The authors found the non-linear ANN outperformed the MLR. However, ANN models have lower accuracy rate for the "potential outbreak" period in comparison to the "no outbreak" period. Population size and date were found to have an excitatory influence (positive ANN weights) upon dengue incidents while previous cases was found to be an inhibitory influence (negative ANN weights). Maximum air temperature seemed to have inconsistent impact on the two population subgroups.

Freeze et al. (2018) expanded their Data Integration and Predictive Analysis System (IPAS) for Influenza like Illness (ILI). Feature engineering was mostly centred on the weekly dengue incidences with normalization of dengue incidence; square and cube of the normalized dengue incidences as nonlinear terms; slope or the change in normalized incidence over 1-4 week horizons for trend analysis. R's caret package was used to create combined models to predict dengue incidences for 1-week and 4-weeks ahead. Features shortlisted from initial Multiple Linear Regression (MLR) were used in the final MLR, Support Vector Machine (SVM), Random Forests (RF) and Boosting models. Significant features found comprised of seasonal

effect (represented by week number), nonlinear terms, trends and regional effect (San Juan vs. Iquitos). RF and Boosting had the smallest and largest testing Mean Square Error (MSE) respectively.

2.5.2 Ensemble Models

Yamana et al. (2016) created three sub-models (F1, F2 and F3) without using the climatic and population data provided in the competition. F1 involved making simplistic assumptions about the distributions of S, I, D, R0 components within the Susceptible–Infectious–Recovered (SIR) model. Next, multiple SIR models was aggregated using Ensemble Adjustment Kalman Filter (EAKF) data-assimilation method. F2 enlisted Bayesian Model Averaging (BMA) in forecasting current week's dengue incidence as the weighted sum of the previous weeks' dengue incidences from the previous seasons. F3 focused on the historical likelihood whereby peak timing was proxy by Gaussian distribution while Gamma distribution was used to generate peak incidence and total incidence in a season. Superensemble (SE) models consisting of different combinations of F1, F2 and F3 were constructed by weighting their respective performance in the preceding weeks using BMA. SEs reported overall lower Mean Absolute Error (MSE) than the individual models. The reliability of F2 among the individual models was carried forward into the outperformance of SE(F1, F2) among the SEs.

Johnson et al. (2018) were one of the six winners of the 2015 Dengue Forecasting Challenge project. The weekly incidences underwent a novel square-logarithmic transformation. hetGP (heteroskedistic Gaussian Process) disregard climatic variables in favour of a phenomenological approach. It overcomes heteroscedasticity by deploying a nonparametric Gaussian Process (GP) regression fitting. The model relies on four predictors (season-time, sine wave, starting level and severity indicator) to match current season to the most similar historical trajectory. Generalized Linear Model (GLM) involved negative binomial with a log link while augmenting given climatic variables (excluding humidity) with Southern Oscillation Index (SOI) and value of El Nino 1/2. The covariates underwent extensive feature engineering which entailed the combination of autoregressive, trend, trigonometric and transformation elements. Monte Carlo simulations were run to generate the forecasts. The hybrid hetGP was among the top three models for San Juan while its performance was less consistent for Iquitos. The authors noticed that the parsimonious F2 model from Yamana et al.

(2016) shared the advantageous historical seasonal pattern memorization and matching capability as their hetGP.

Sougata, Acebedo and Chua (2017) performed feature engineering with new variables from Mosquito lifecycle related lagged effects, up to 32 weeks, interaction between climatic variables and forecasted dengue cases from decomposition-based time series models. An ensemble model took the median prediction from the following sub-models, namely MLR, WMLR and CPM. MLR used overall data which enabled San Juan and Iquitos to learn from each other's dengue experience. Weighted Multiple Linear Regression (WMLR) emphasized more on the epidemic periods. Covariate Pattern-Matching (CPM) relied on the closest matching historical window of each 19 raw climatic variables to predict the daily case changes. CPM overfitted the training data as its testing dataset's MAD of 26.05% was much larger than training set's (at 7.23%). The ensemble model outperformed all sub-models, with the exception for CPM in the training set. Significant coefficients from MLR and WMLR models indicated that vegetation and precipitation had negative impact while humidity and maximum temperature had positive influence on dengue occurrence.

Buczak et al. (2018) was another winning team from the 2015 challenge. Their ensemble algorithm weighted the top 300 performing models from among the following by comparing their forecast errors in the past four years: (1) 1248 distinct additive seasonal Holt-Winters (HW) models (2) Two-dimensional Method of Analogues models which relied on historical sequence of precipitation with 2-weeks lag to forecast the weekly dengue incidences for 4-52 weeks in advance. (3) Simple historical models of peak height, peak weak and total cases. The authors' team was the strongest contender for Iquitos' seasonal peak height and total cases but their San Juan predictions were mixed.

2.5.3 Gaps identified within literature review

According to National Oceanic and Atmospheric Administration (2018a), with the exception of dengue in the study locations or nearby locations, participants are allowed to augment the model with exogenous data sources, such as social media and demography. However, none of them utilized internet searches as exogenous regressor.

Furthermore, only one of the six papers published (Freeze, Erraguntla and Verma, 2018) adopted forward looking nowcasting (forecasting the present or into the near future) which is

most practical for situational awareness (Marques-Toledo et al., 2017). Buczak et al. (2018) adopted nowcasting in one of the sub-model but not at the ensemble model level.

Meanwhile, the dengue forecasting models which incorporated GTs was not built for nowcasting nor take climatory conditions into consideration (Yang et al, 2017). Hence, GTs can be utilized as external regressor to supplement the official DengAI climatic dataset to nowcast dengue cases in the two cities under a LASSO framework.

3. Methodology

3.1 Data Collection

DrivenData (2018b) provides the climatic variables but excluded dengue cases from test dataset. NOOA provides the annual population data for the two cities. The weekly frequency of English and Spanish dengue query terms from Yang et al. (2017) was extracted via Google Trends API.

Table 3.1: Query terms to be extracted from Google Trend API from the two cities

Spanish (from Mexico)	English (from Singapore)
Dengue dengue.dengue.dengue el.dengue dengue.sintomas sintomas.del.dengue dengue.hemorragico sintomas.de.dengue que.es.dengue dengue.clasico dengue.mosquito	dengue dengue.fever dengue.symptoms symptoms.dengue.fever symptoms.of.dengue dengue.mosquito mosquito

3.2 Preprocessing

The annual population was changed from annual to daily frequency using upsampling (Brownlee, J., 2016) and interpolation techniques (The SciPy community, 2018; The pandas project, 2018a). San Juan has an outlier in 1999 and population decline after 2005.

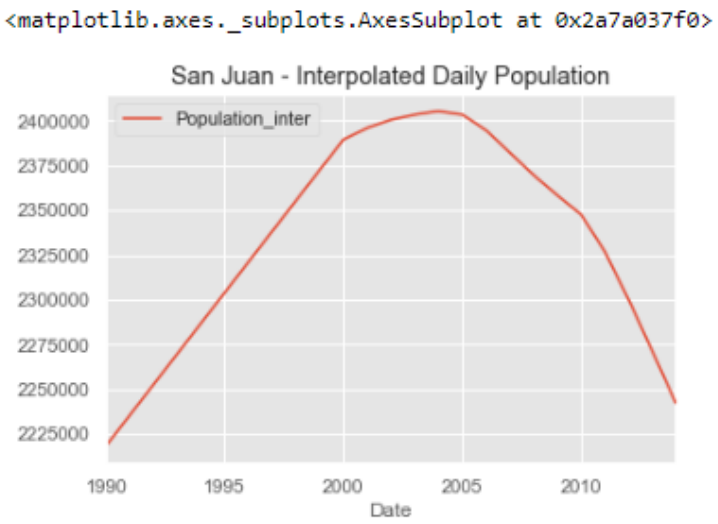


Figure 3.1: Interpolated daily population of San Juan

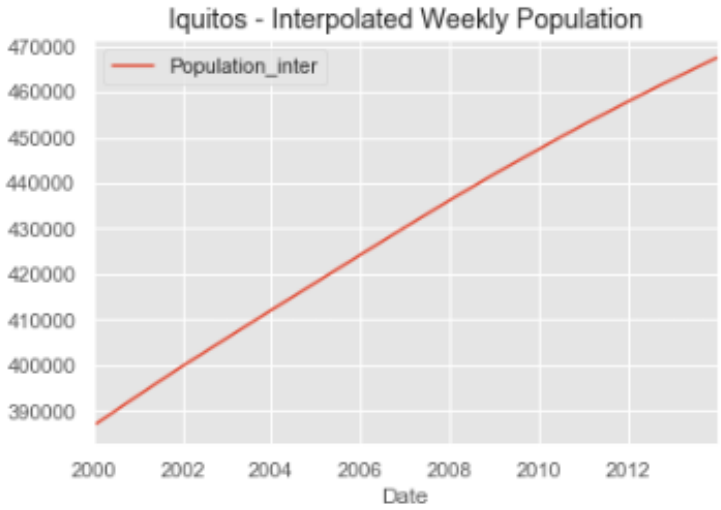


Figure 3.2: Interpolated daily population of Iquitos

Population data was inner joined with DrivenData's datset using 'City_week_start_date' as unique identifier. total_cases was normalized to be cases per 100,000 population under 'normal_total_cases'. 'Advance4W_normal_total_cases' carried forward 'normal_total_cases' 4 weeks in advance. All temperature related variables were standardized using Celcius.

3.3 Explorative Data Analysis (EDA)

San Juan is more urbanized with less rainfall, less humid, higher temperature and less critical dengue situation based on descriptive statistics.

Descriptive statistics of sj_train

	year	weekofyear	ndvi_ne	ndvi_nw	ndvi_se	ndvi_sw	precipitation_amt_mm	reanalysis_precip_amt_kg_per_m2	reanalysis_relative_humidity_percent
count	932.00	932.00	741.00	884.00	913.00	913.00	923.00	926.00	926.00
mean	1998.79	26.55	0.06	0.07	0.18	0.17	35.60	30.55	78.58
std	5.19	15.04	0.11	0.09	0.06	0.06	44.66	35.67	3.39
min	1990.00	1.00	-0.41	-0.46	-0.02	-0.06	0.00	0.00	66.74
25%	1994.00	13.00	0.01	0.02	0.14	0.13	0.00	10.90	76.27
50%	1999.00	27.00	0.06	0.07	0.18	0.17	20.96	21.35	78.67
75%	2003.00	40.00	0.11	0.12	0.21	0.20	52.32	37.08	80.98
max	2008.00	53.00	0.49	0.44	0.39	0.38	390.60	570.50	87.58

8 rows × 24 columns

Figure 3.3: Descriptive statistics of sj_train

Descriptive statistics of iq_train

	year	weekofyear	ndvi_ne	ndvi_nw	ndvi_se	ndvi_sw	precipitation_amt_mm	reanalysis_precip_amt_kg_per_m2	reanalysis_relative_humidity_percent
count	516.00	516.00	513.00	513.00	513.00	513.00	512.00	512.00	512.00
mean	2004.96	26.53	0.26	0.24	0.25	0.27	64.22	57.36	88.61
std	2.90	15.09	0.08	0.08	0.08	0.09	35.34	50.04	7.60
min	2000.00	1.00	0.06	0.04	0.03	0.06	0.00	0.00	57.79
25%	2002.00	13.00	0.20	0.18	0.19	0.20	39.09	23.95	84.24
50%	2005.00	27.00	0.26	0.23	0.25	0.26	60.47	46.22	90.89
75%	2007.00	40.00	0.32	0.29	0.30	0.33	85.76	71.07	94.56
max	2010.00	53.00	0.51	0.45	0.54	0.55	210.83	362.03	98.61

8 rows × 24 columns

Figure 3.4: Descriptive statistics of iq_train

San Juan has the worst dengue outbreak in October-November 1994. Iquitos has one particularly serious outbreak in December 2004.

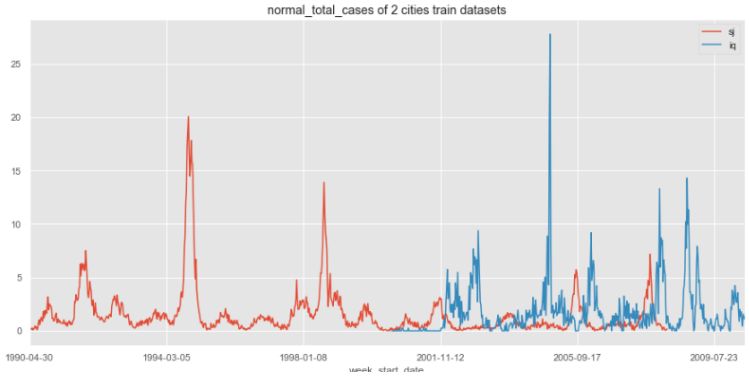


Figure 3.4: normal_total_cases of 2 cities train datasets

'weekofyear' should be included as seasonality predictor as both cities have different peak and low seasons.

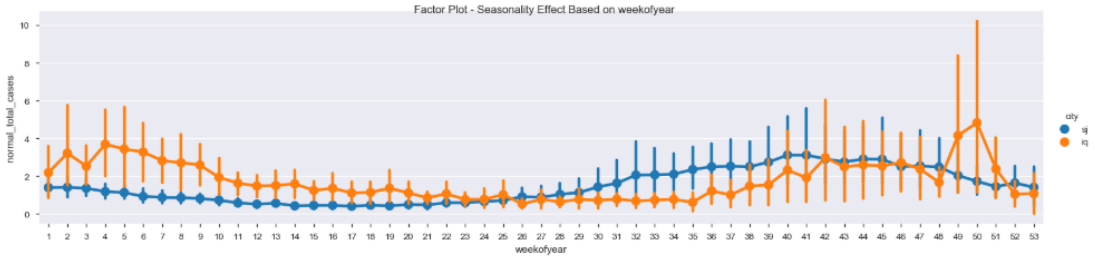


Figure 3.5: Factor Plot by city for ‘weekofyear’ vs. ‘normal_total_cases’

Critical sources of missing values are San Juan's ndvi, esp. ndvi_ne and Iquitos' weather station, esp. maximum & diural range temperatures. No variable was excluded as none has more than 50% missing. Single imputation of Multiple Imputation by Chained Equations (MICE) was used based on Missing values are Missing At Random (MCAR) pattern.

The climatic variables from different measurement systems are compared by side by side. The different distributions of normal_total_cases and climatic variables for the 2 cities warrant separate nowcasting models.

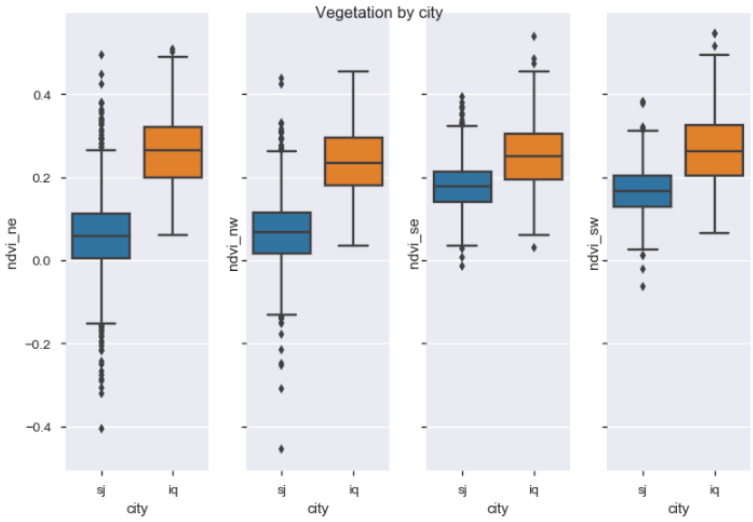


Figure 3.6: Vegetation by city

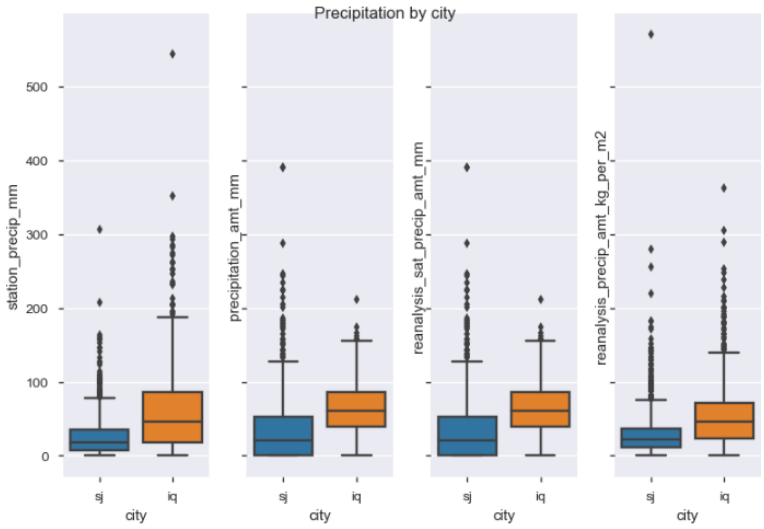


Figure 3.7: Precipitation by city

Multicollinearity is more profound among San Juan's variables than Iquitos'. Pearson correlation identified variables with correlation closed to one for removal. The climatic variables have stronger impact in San Juan than Iquitos, albeit overall weak correlations.

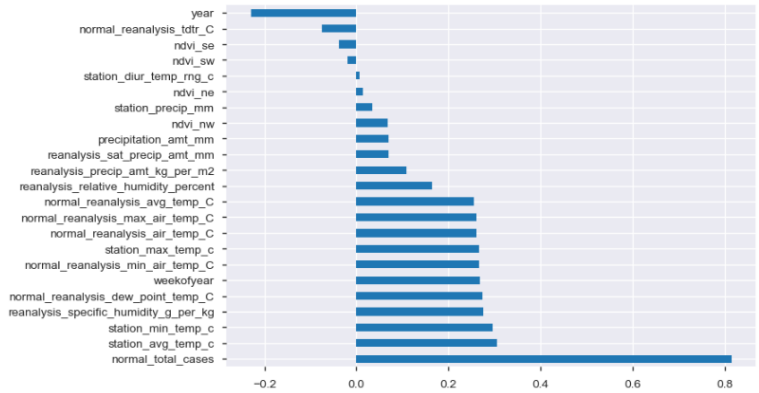


Figure 3.8: Correlations between 'Advance4W_normal_total_cases' vs. predictors – sj_train

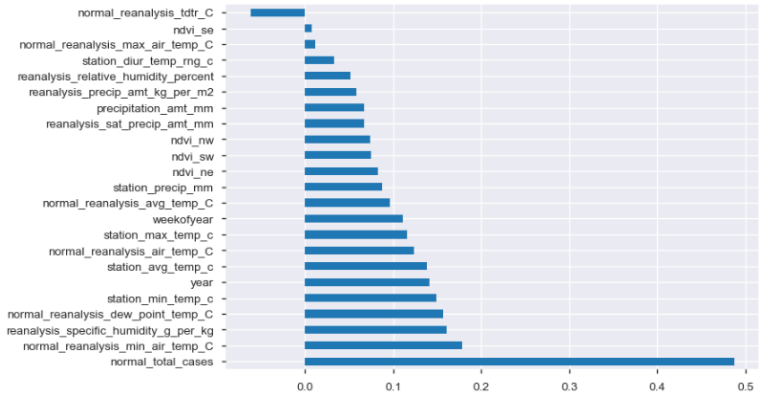


Figure 3.9: Correlations between 'Advance4W_normal_total_cases' vs. predictors – iq_train

The right-skewed variables with skewness more than +2 underwent ln(1+x) transformation to reduce the skewness, followed by standardization. No feature was further removed based on 0.7 variance threshold.

Cross Correlation Function (CCF) was performed in R to determine the lag effects of climatic variable up to 32 weeks. 364 and 241 new features were added to sj_train and iq_train respectively.

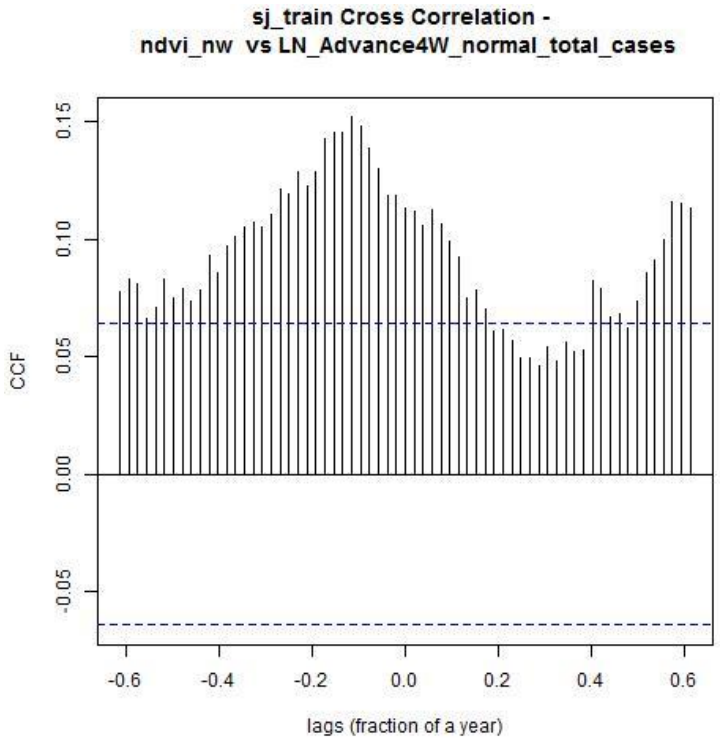


Figure 3.10: CCF and 95% confidence interval for ndvi_nw

3.4 Modeling

Two Lasso was constructed for each city to evaluate the value added from feature engineering. The initial 32 weeks with NaN from lagged climatic variables were deleted in order to standardize between the base and feature engineering models. The carrying forward of total_cases by four weeks led to four additional weeks in the test dataset. The first 70% of train dataset was assigned as subtrain dataset while the last 30% were assigned as subvalid datasets.

Table 3.2: Four Lasso models and data splits

City	Base model with no feature engineering	Model with feature engineering from lagged climatic variables	Subtrain	Subvalid	Test
San Juan	M1	M2	10-Dec-1990 to 15-Jan-2003	22-Jan-2003 to 25-Mar-2008	1-Apr-08 to 23-Apr-13
Iquitos	M3	M4	12-Feb-2001 to 6-Aug-2007	13-Aug-2007 to 28-May-2010	4-Jun-10 to 25-Jun-13

4. Results/ Analysis

4.1 San Juan M1 base model

Lasso coefficient path in Figure 4.1 check the impact of different alphas on predictors' coefficients. M1's initial coefficients ranges from (-1.25 to +0.75). The magnitude of coefficients decreases as alpha increases and zerolized as alpha approaches 1.

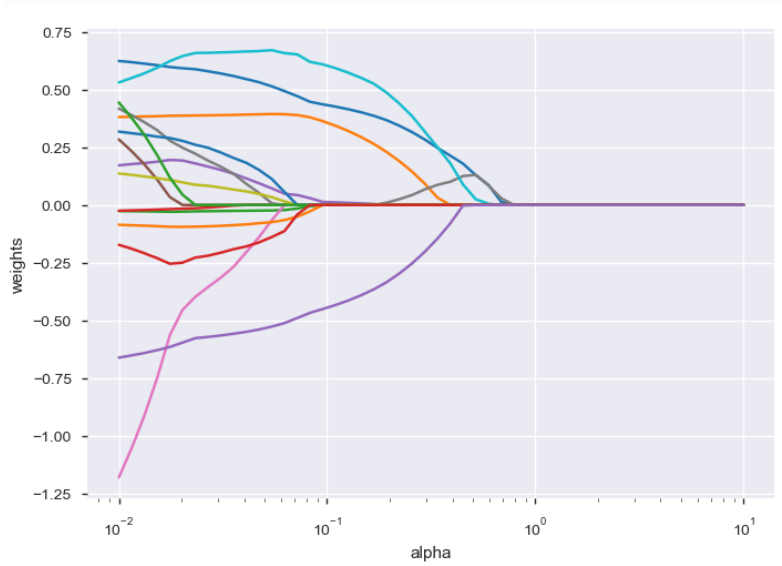


Figure 4.1: M1 - Lasso Coefficient Path for San Juan

GridSearchCV with 10-splits from TimeSeriesSplit chose the best alpha, refitted the model and compute the associated subvalid error. Initial grid search return optimal alpha = 1.072267222010323, where the negative MAE chart peak. However, this corresponds to weight = 0 in Figure 4.1. This will cause model with insignificant coefficients and zero predictions which are not useful at all.

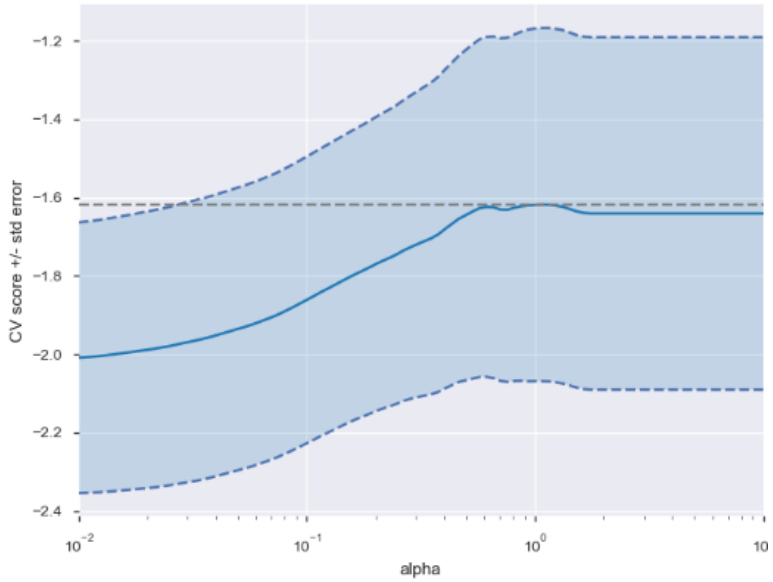


Figure 4.2: M1 - initial grid search for San Juan

Therefore, alpha range was refined to 0.01-0.1 where there are at least a handful of non-zero weights. According to Figure 4.3, an unique solution was found at alpha = 0.1 and MAE improved from subtrain 1.3401 to subvalid 1.1104.

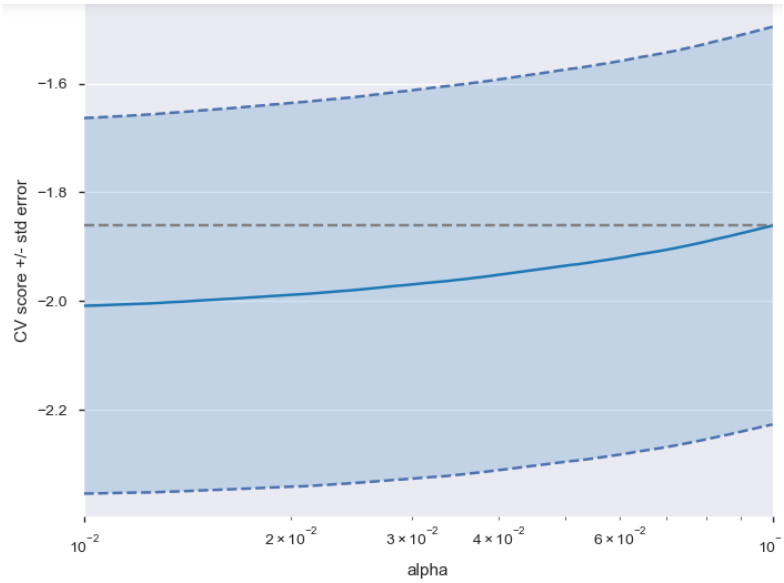


Figure 4.3: M1 - refined grid search for San Juan

According to Figure 5.2.5C, 5/15 predictors were found to be significant in predicting Advance4W_normal_total_cases with non-zero coefficients. They are

- station_max_temp_C - the hotter, the more cases.
- normal_reanalysis_tdtr_C - the bigger the gap between maximum & minimum temperatures, the lesser cases.
- weekofyear - cases increase toward year end.
- ndvi_nw - the more vegetation on the north-west direction, the more cases.
- LN_renalaysis_precip_amt_kg_per_m2 - the more rainfall, the more cases

```
Non-zero coefficient in descending absolute value order:
                                    Coef
station_max_temp_c                 0.604519
normal_reanalysis_tdtr_C          -0.445335
weekofyear                         0.433175
ndvi_nw                            0.357090
LN_reanalysis_precip_amt_kg_per_m2 0.012586
```

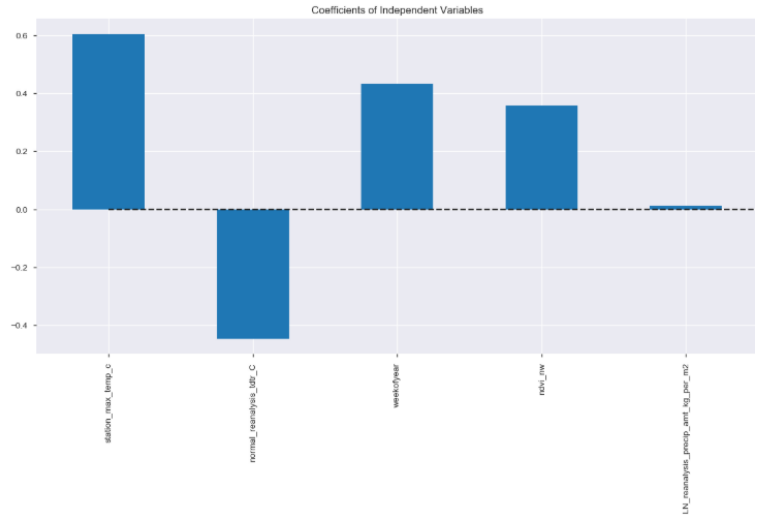


Figure 4.4: M1 - bar chart of predictors in descending order of absolute coefficients

The predictions of M1 models using subtrain, subvalid and test datasets were merged and compared against the actual total_cases in Figure 4.5.

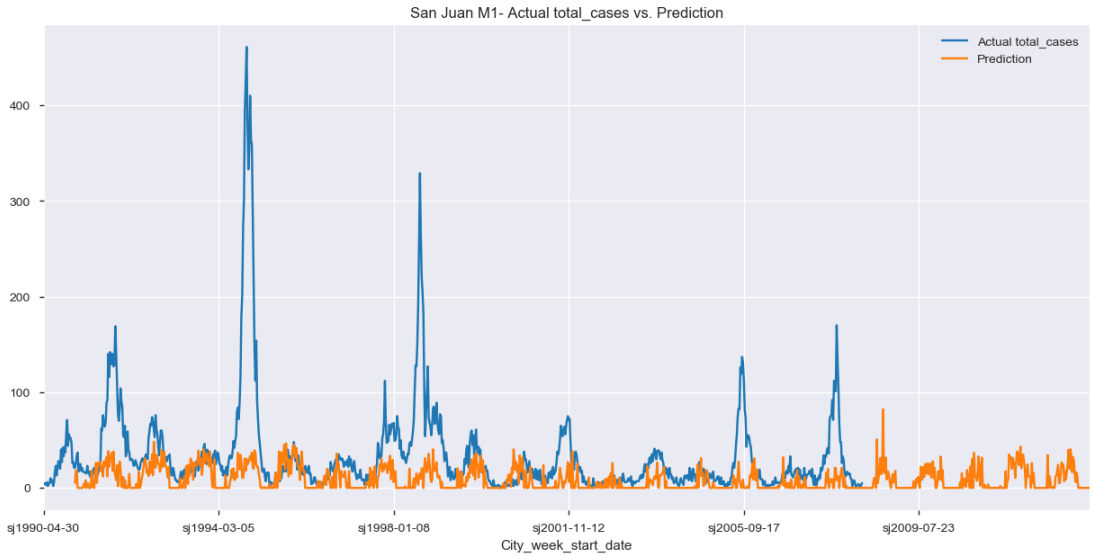


Figure 4.5: M1 model – predicted vs. actual total_cases

4.2 San Juan M2 feature engineering model

The Lasso coefficient path in Figure 4.6 demonstrated that M2' initial coefficients ranges from (-0.5 to +0.7).

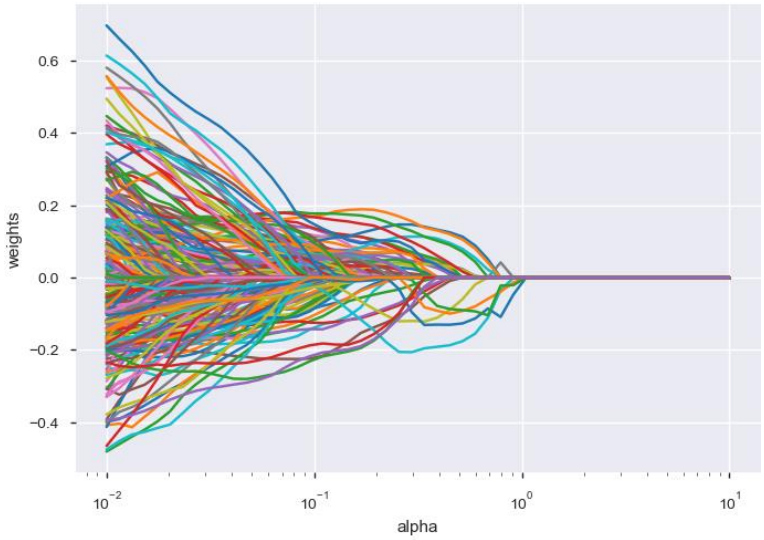


Figure 4.6: M2 model – Lasso coefficient path

Initial grid search in Figure 4.7 found the optimal alpha = 1.3894954943731375, where the negative MAE chart peak. However, this corresponds to weight = 0 in Figure 4.6.

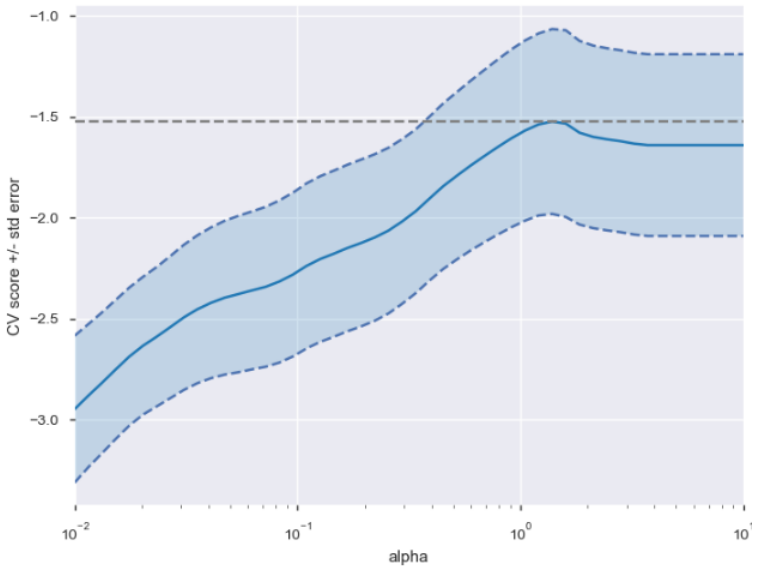


Figure 4.7: M2 model – Initial grid search

Alpha range was fined tune 10E-0.75 to 10E-0.35 where there are at least 5 sets of engineered features with non-zero weights. According to Figure 4.8, unique solution was found at alpha = 0.5623413251903491 and MAE improved from subtrain 1.6195 to subvalid 1.1781.

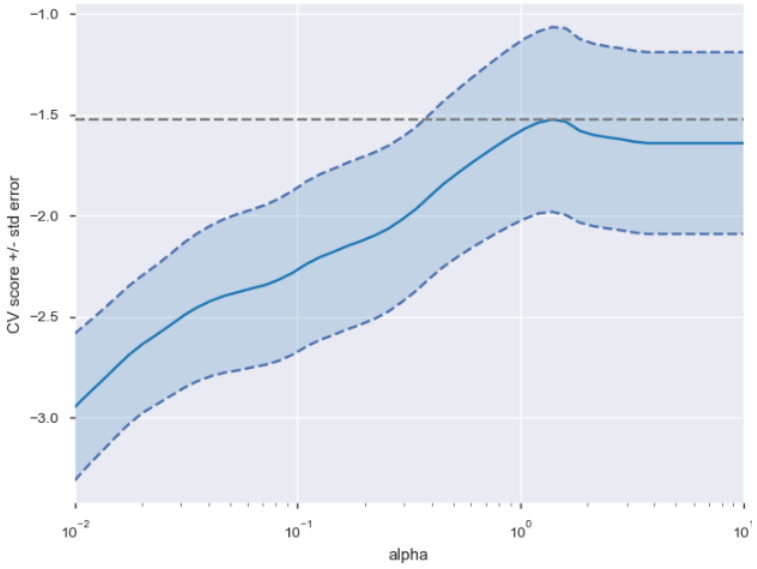


Figure 4.8: M2 model – Initial grid search

According to Figure 4.9, 19/365 predictors were found to be significant in predicting Advance4W_normal_total_cases with non-zero coefficients. The top 5 significant predictors are

- FE_normal_reanalysis_max_air_temp_C from 28-32 weeks ago - the hotter, the lower cases.
- FE_station_diur_temp_rng_c from 23, 26-32 weeks ago - the bigger the gap between maximum & minimum temperatures, the more cases.
- FE_ndvi_nw of 28, 30-32 weeks ago - the higher the vegetation at north-west direction, the more cases.
- FE_station_min_temp_c of 6 weeks ago - the higher the minimum ground temperature, the more cases
- FE_LN_precipitation_amt_mm of 32 weeks ago - the more rainfall, the lesser the cases.

Non-zero coefficient in descending absolute value order:

	Coef
FE_normal_reanalysis_max_air_temp_C(t-29)	-0.187649
FE_normal_reanalysis_max_air_temp_C(t-30)	-0.130869
FE_station_diur_temp_rng_c(t-29)	0.125769
FE_station_diur_temp_rng_c(t-30)	0.120828
FE_normal_reanalysis_max_air_temp_C(t-31)	-0.098969
FE_station_diur_temp_rng_c(t-28)	0.096412
FE_station_diur_temp_rng_c(t-31)	0.091030
FE_normal_reanalysis_max_air_temp_C(t-28)	-0.090543
FE_normal_reanalysis_max_air_temp_C(t-32)	-0.087704
FE_station_diur_temp_rng_c(t-32)	0.040954
FE_ndvi_nw(t-31)	0.034419
FE_station_diur_temp_rng_c(t-26)	0.033460
FE_station_diur_temp_rng_c(t-27)	0.030560
FE_ndvi_nw(t-30)	0.026436
FE_station_min_temp_c(t-6)	0.017392
FE_ndvi_nw(t-32)	0.012119
FE_station_diur_temp_rng_c(t-23)	0.011349
FE_LN_precipitation_amt_mm(t-32)	-0.010362
FE_ndvi_nw(t-28)	0.007760

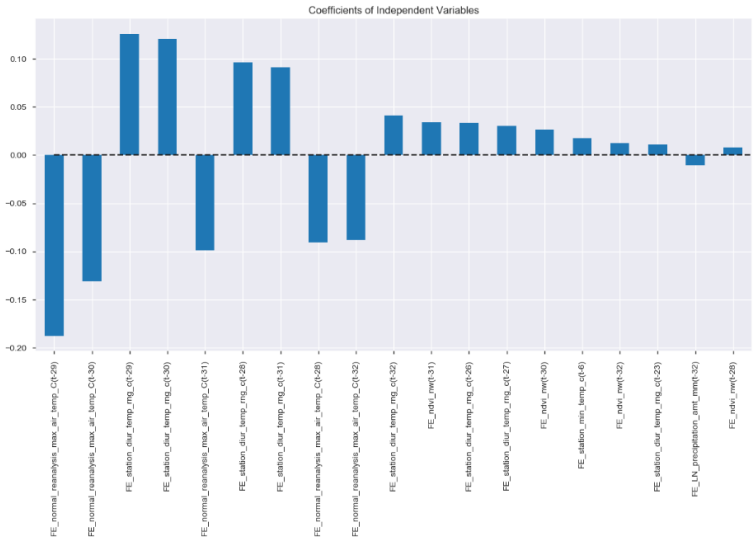


Figure 4.9: M2 model – Lasso coefficients

The predictions of M2 models using subtrain, subvalid and test datasets were plotted against the actual total_cases in Figure 4.10.

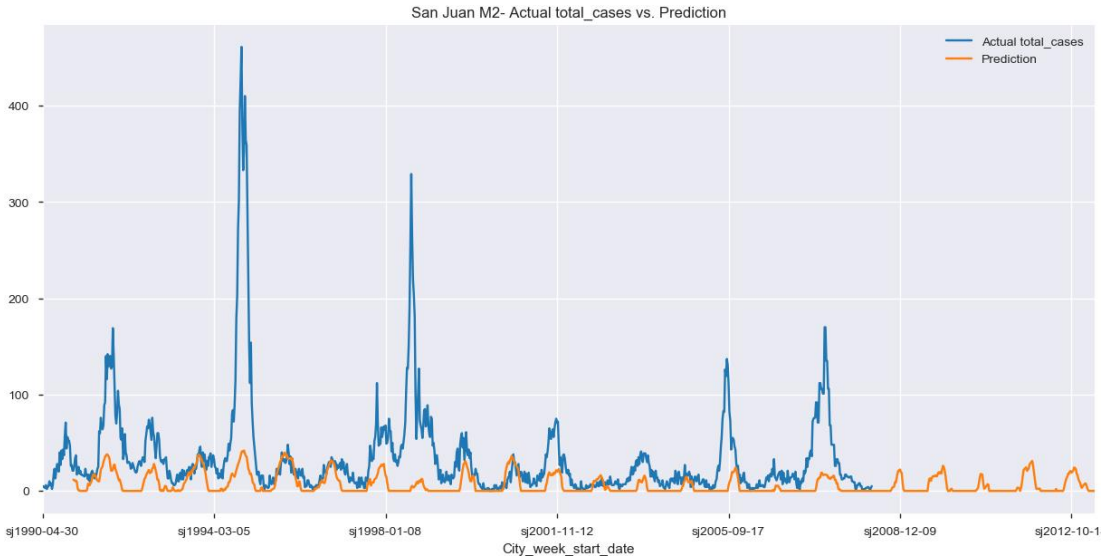


Figure 4.10: M2 model – predicted vs. actual total_cases

4.3 Train Iquitos' M3 base model

Lasso coefficient path in Figure 4.11 showed that M3's initial coefficients ranges from (-0.6 to +0.6).

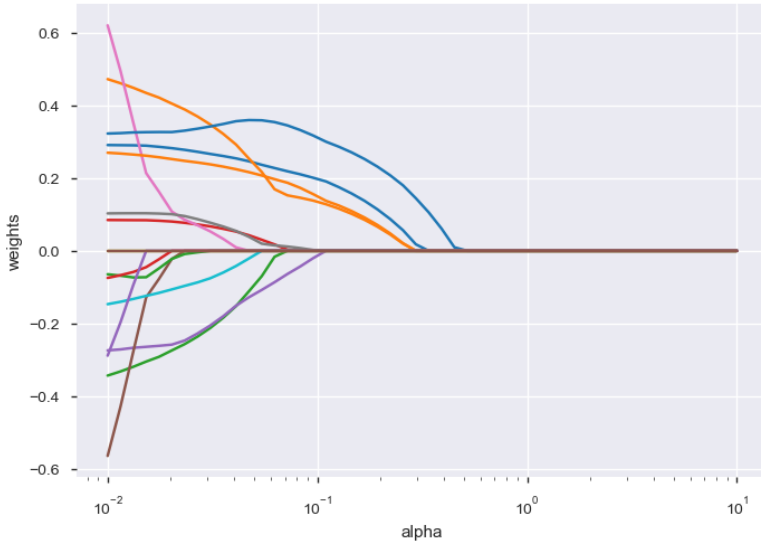


Figure 4.11: M3 – Lasso coefficient path

The initial grid search Figure 4.12: M3 recorded no optimal solution as the negative MAE chart peaked and flatten for alpha => 0.7564633275546291. This corresponds to weight = 0 in Figure 4.11. This will cause model with insignificant coefficients and zero predictions which are not useful at all.

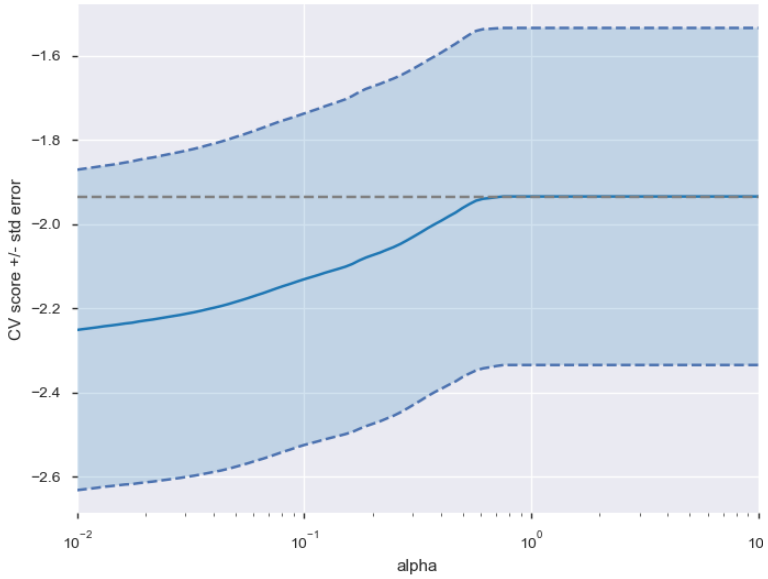


Figure 4.12: M3 – initial grid search

Therefore, we need to fine tune the alphas to be the range of 0.01-0.1 where there are at least a handful of non-zero weights. According to Figure 4.13, an unique solution was found at alpha = 0.1 while MAE deteriorated from subtrain 1.7303 to subvalid 2.4149.

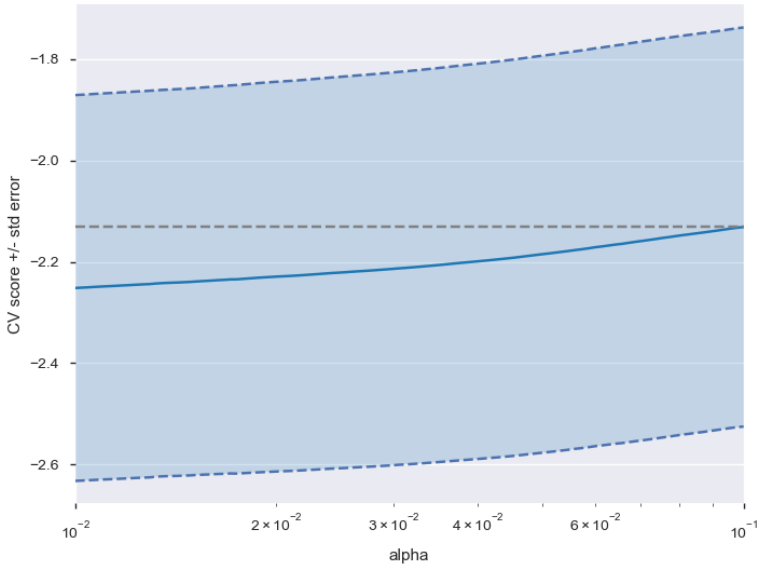


Figure 4.13: M3 – refined grid search

According to Figure 4.14, 6/15 predictors were found to be significant in predicting Advance4W_normal_total_cases with non-zero coefficients. They are

- station_min_temp_c - the hotter, the more cases.
- weekofyear - cases increase toward year end.
- LN_station_precip_mm - the more rainfall, the more cases.
- ndvi_ne - the more vegetation on the north-east direction, the more cases.
- LN_reanalysis_precip_amt_kg_per_m2 - the less rainfall, the more cases

Non-zero coefficient in descending absolute value order:

	Coef
station_min_temp_c	0.312148
weekofyear	0.197789
LN_station_precip_mm	0.149914
ndvi_ne	0.135542
LN_reanalysis_precip_amt_kg_per_m2	-0.012013
station_avg_temp_c	0.000366

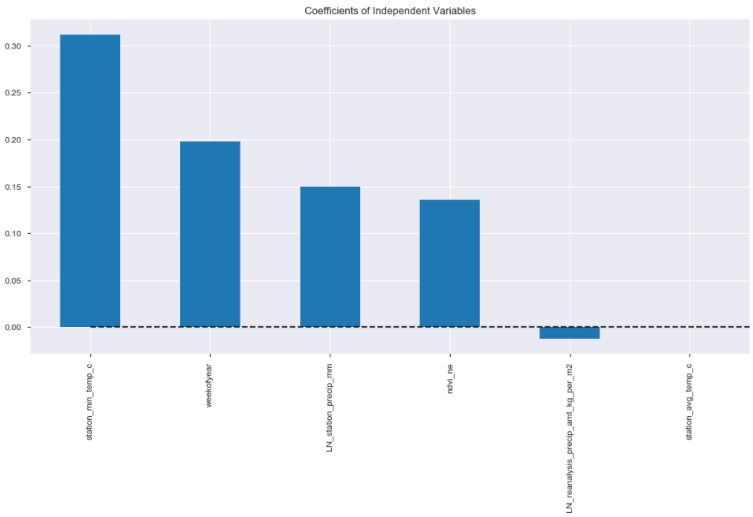


Figure 4.14: M3 – Lasso coefficients

The predictions of M2 models using subtrain, subvalid and test datasets were plotted against the actual total_cases in Figure 4.15.

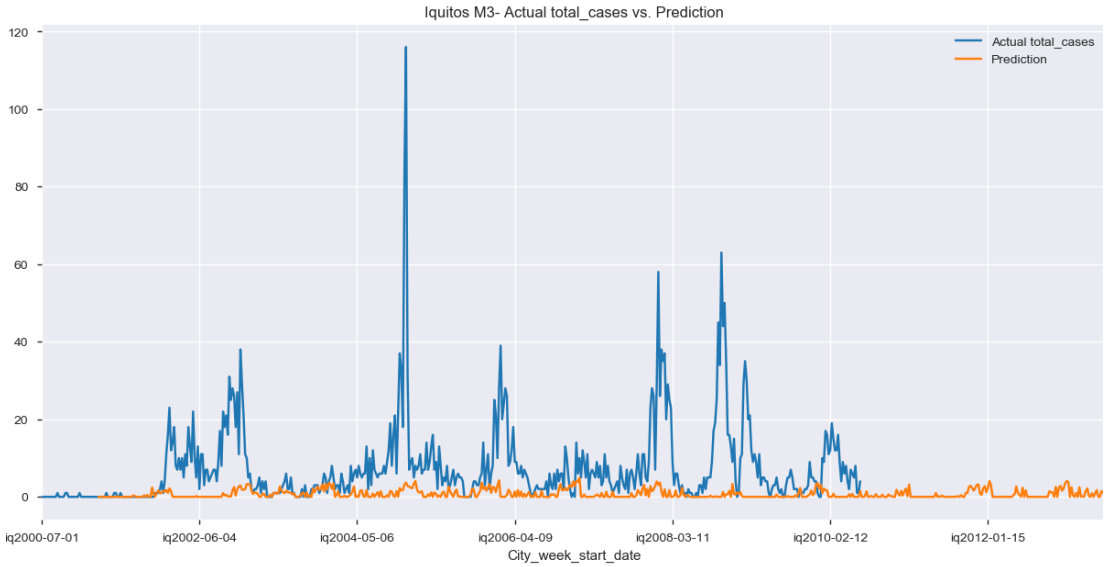


Figure 4.15: M3 – predicted vs. actual total_case

4.4 Train Iquitos' M4 feature engineering model

The Lasso coefficient in Figure 4.16 demonstrated thatM4's initial coefficients ranges from (-1.0 to +1.30).

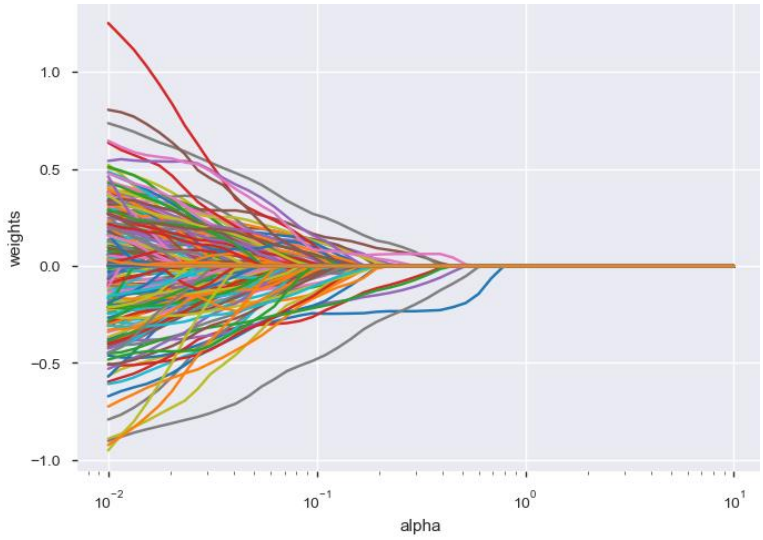


Figure 4.16: M4 – Lasso coefficient path

Initial grid search in Figure 4.17 found no unique solition negative MAE chart peak and plateaued for alpha => 1.2067926406393288. However, this corresponds to weight = 0 in Figure 4.16. This will cause model with insignificant coefficients and zero predictions which are not useful at all.

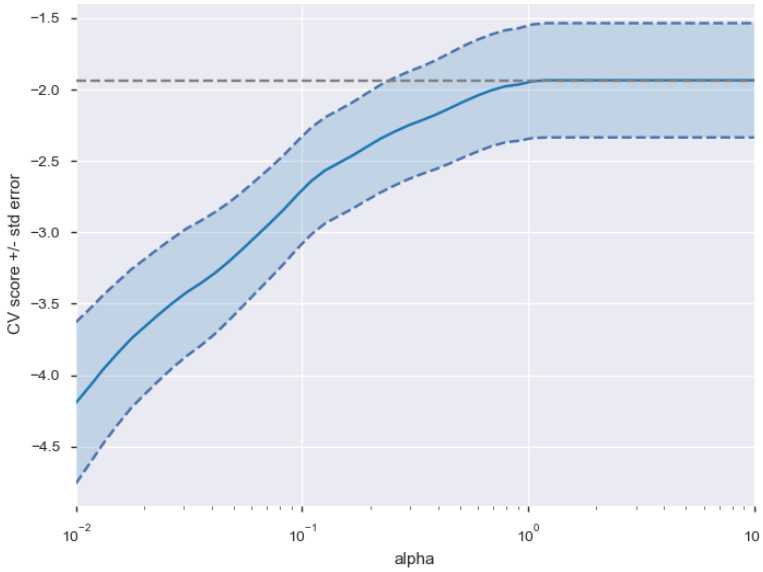


Figure 4.17: M4 – Initial grid search

Alpha range was fined tune to 10E-0.75 - 10E-0.35 where there are at least 5 sets of engineered features with non-zero weights. According to Figure 4.18, unique solution was found at alpha = 0.31622776601683794 and MAE deteriorated from subtrain 1.7741 to subvalid 1.1781.

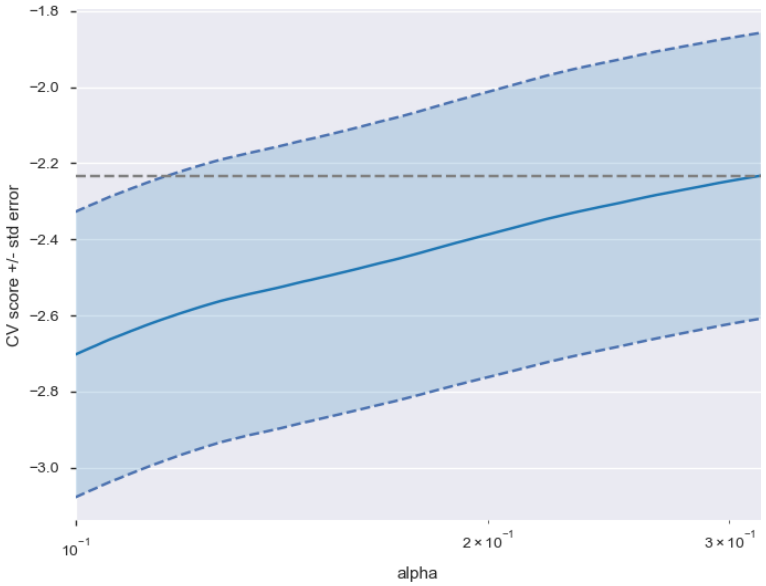


Figure 4.18: M4 model – Refined grid search

According to Figure 4.19, 8/264 predictors were found to be significant in predicting Advance4W_normal_total_cases with non-zero coefficients. The top 5 significant predictors are

- FE_LN_reanalysis_precip_amt_kg_per_m2 from 13 weeks ago - the more rainfall, the lesser cases.
- FFE_normal_reanalysis_min_air_temp_C from 13 weeks ago - the higher the minimum temperatures, the more cases.
- FE_station_avg_temp_c of 19-20 weeks ago - the hotter the ground average temperature, the lesser cases
- FE_normal_reanalysis_tdtr_C from 11 weeks ago - the larger the temperature gap, the more cases.
- FE_station_min_temp_c of 0, 2, 23 weeks ago - the higher the minimum ground temperature, the more cases.
- FE_LN_precipitation_amt_mm of 32 weeks ago - the more rainfall, the lesser the cases.

```
Non-zero coefficient in descending absolute value order:
                                                Coef
FE_LN_reanalysis_precip_amt_kg_per_m2(t-13) -0.231969
FE_normal_reanalysis_min_air_temp_C(t-13)   -0.183084
FE_station_avg_temp_c(t-20)                 -0.096481
FE_normal_reanalysis_tdtr_C(t-11)            0.061924
FE_station_min_temp_c(t-23)                 -0.059650
FE_station_min_temp_c(t-2)                   0.057043
FE_station_min_temp_c(t)                     0.049160
FE_station_avg_temp_c(t-19)                 -0.047247
```

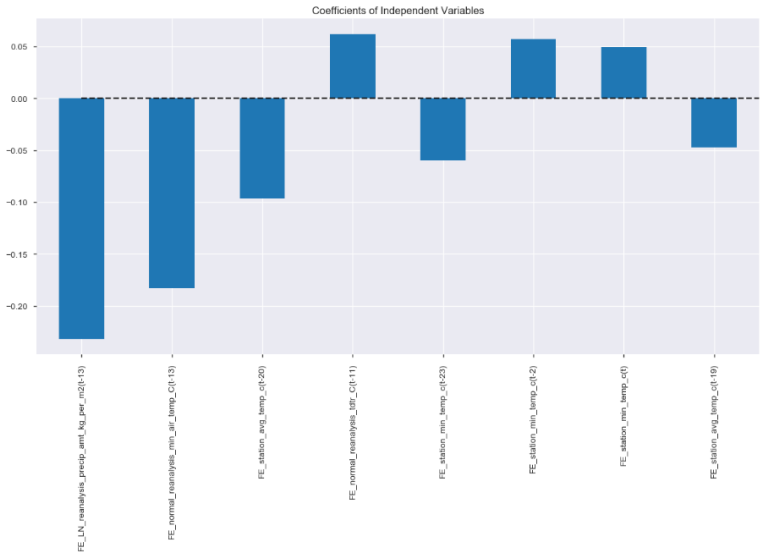


Figure 4.19: M4 – Lasso coefficients

The predictions of M4 models using subtrain, subvalid and test datasets were plotted against the actual total_cases in Figure 4.20.

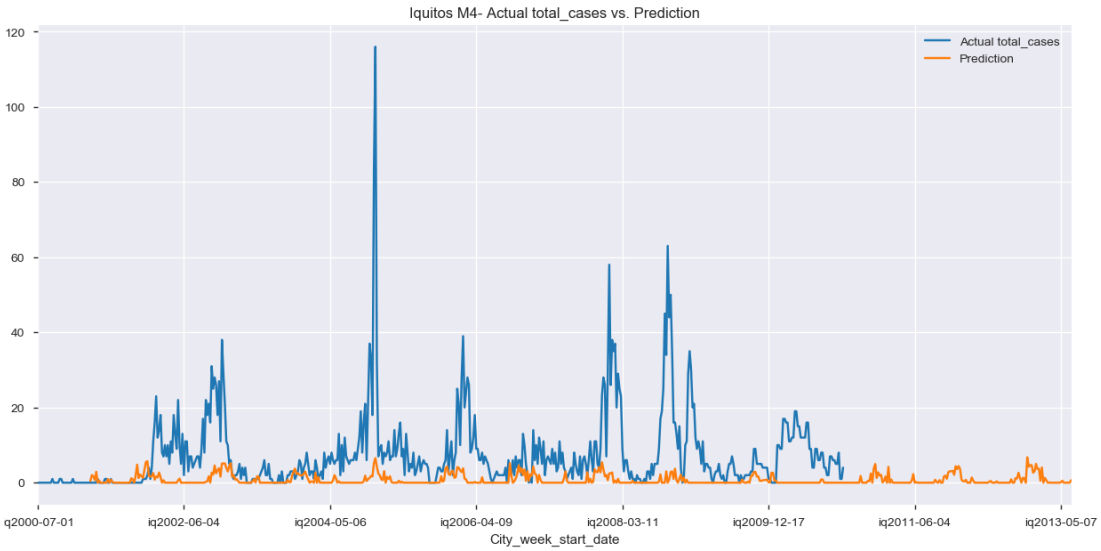


Figure 4.20: M2 model – predicted vs. actual total_cases

4.6 Model Diagnostics

Residual from San Juan's M1 base model demonstrated a right-skewed, non-zero mean and failure to capture the peaks during outbreak. Residual autocorrelation plot (Figure 4.23) did not mimic a white noise pattern since there are significant correlation about 150, 200, 350, 550, 650 lags, which could be some cyclical effect.

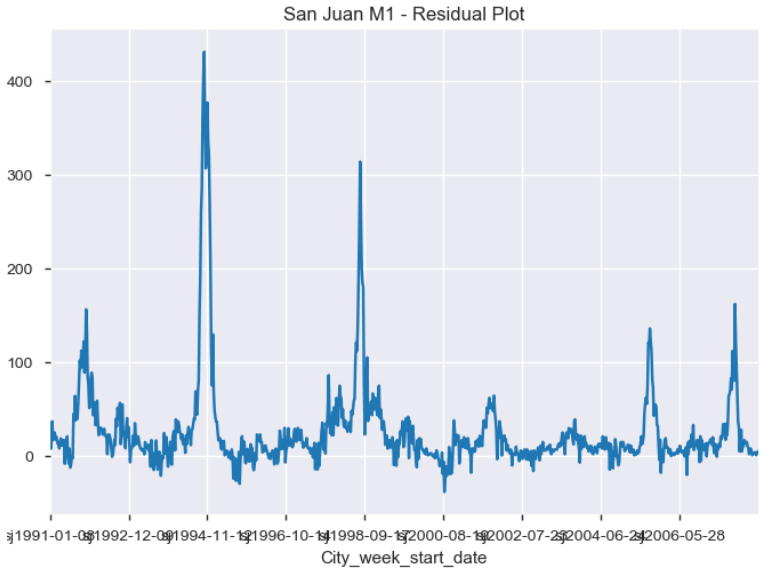


Figure 4.21: M1 – residual line plot

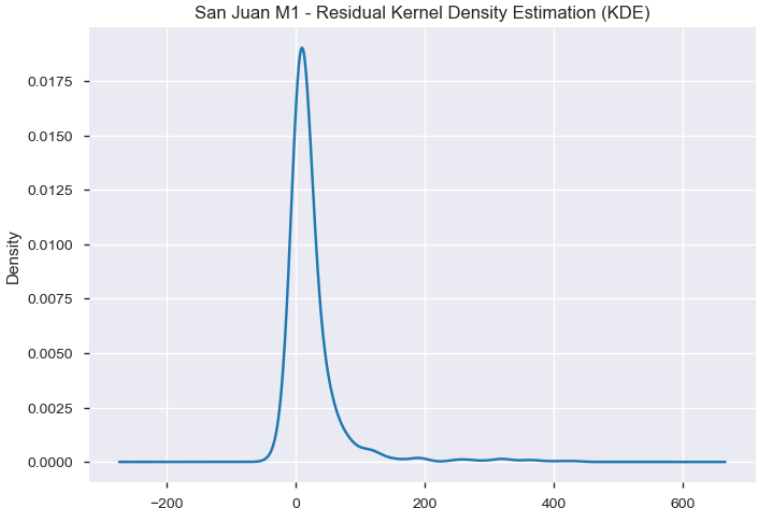


Figure 4.22: M1 – residual KDE

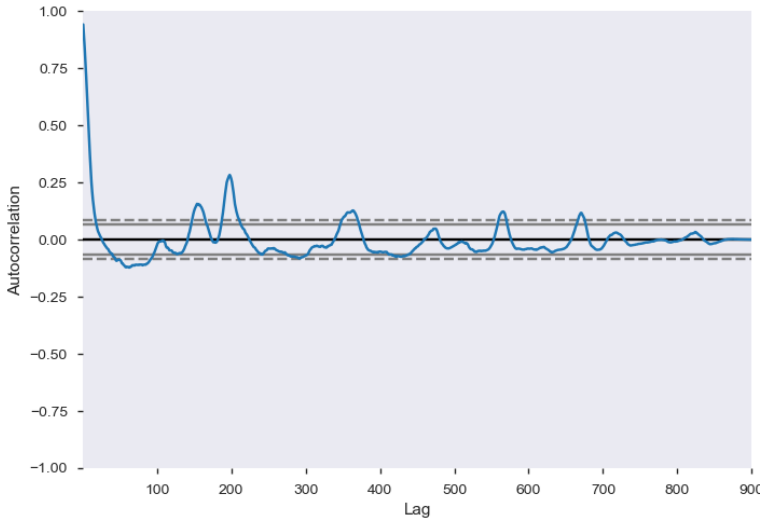


Figure 4.23: M1 – residual ACF plot

Residual from San Juan's M2 feature engineering model demonstrated similar patterns as M1. Residual autocorrelation plot (Figure 4.26) did not mimic a white noise pattern since there are significant correlation about 150, 200, 350, 550, 700 lags, which could be some cyclical effect.

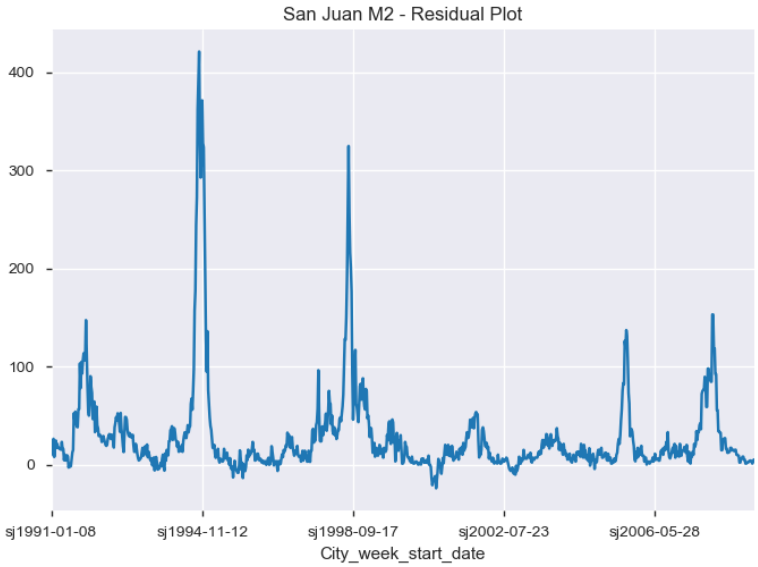


Figure 4.24: M2 – residual line plot

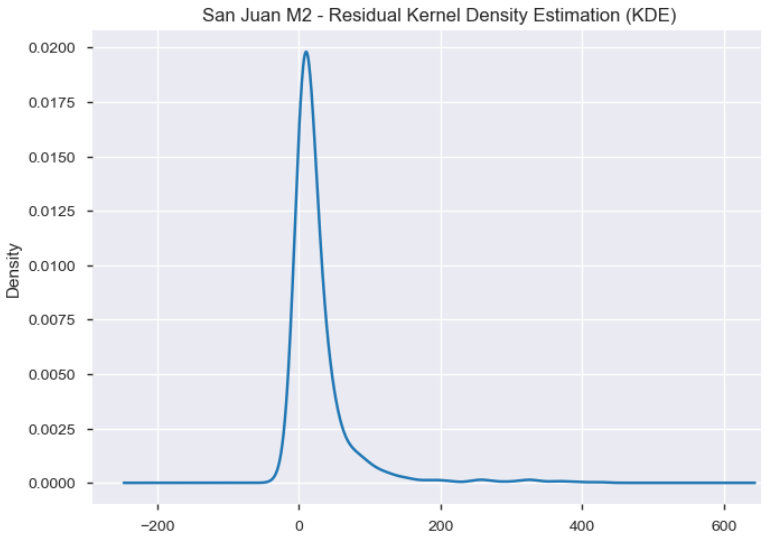


Figure 4.25: M2 – residual KDE

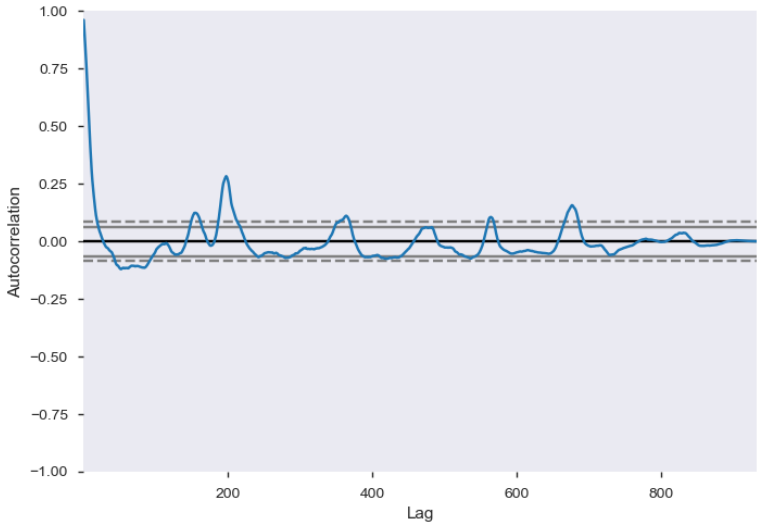


Figure 4.26: M2 – residual ACF plot

Residual from Iquitos' M3 base model demonstrated similar patterns as M1. However, its outliers are not as large as San Juan's. There could be less prominent cyclical effect than San Juan since significant autocorrelations were only noted at about 100, 200 lags (Figure 4.29).

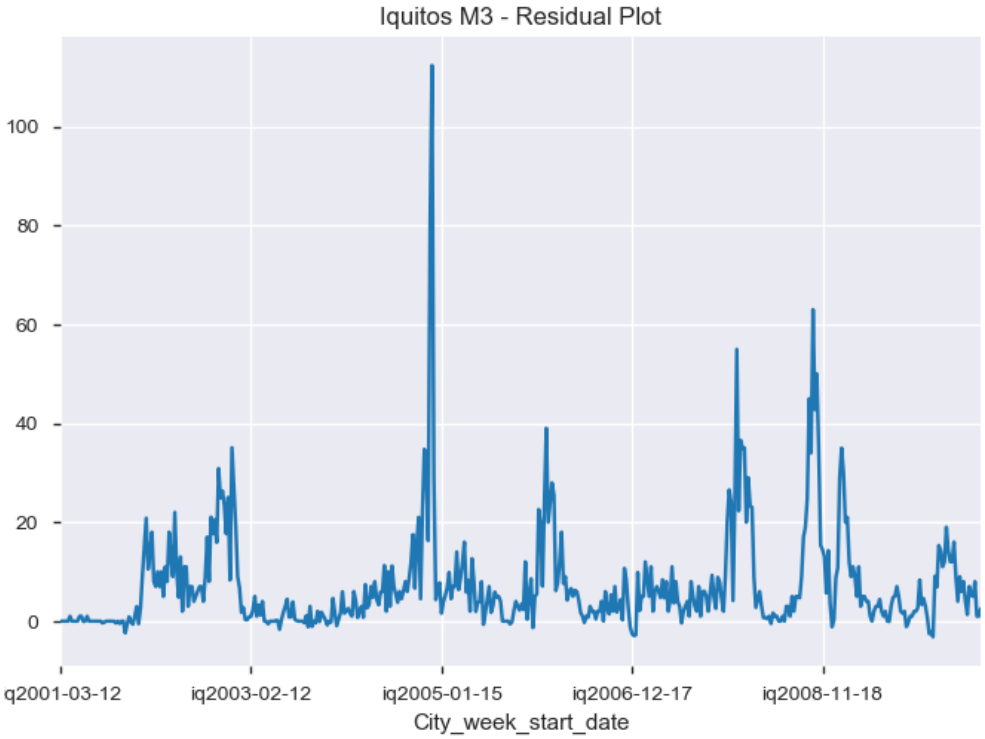


Figure 4.27: M3 – residual line plot

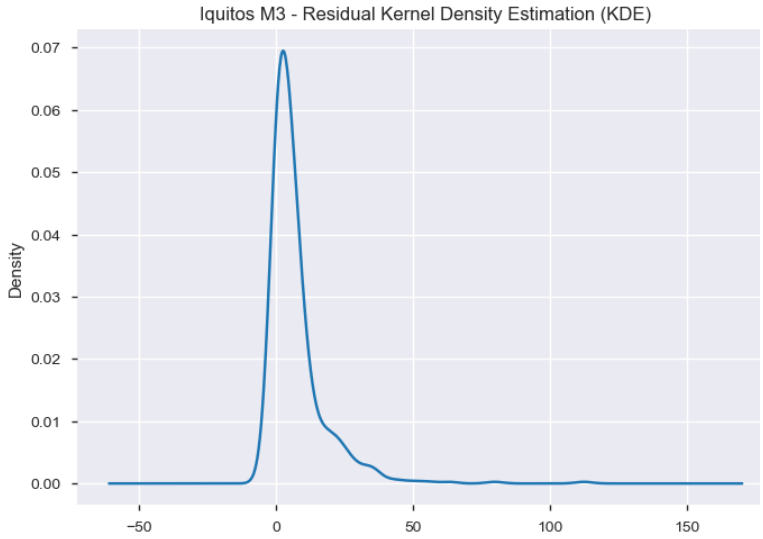


Figure 4.28: M3 – residual KDE

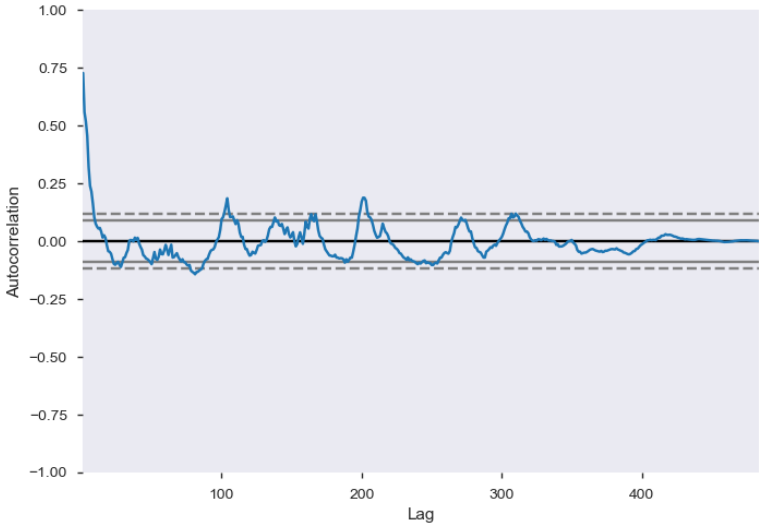


Figure 4.29: M3 – residual ACF plot

Residual from Iquitos' M4 feature engineering model demonstrated similar patterns as M3. Residual autocorrelation plot (Figure 4.32) has only significant autocorrelation about 200 lags, signalling even less cyclical effect.

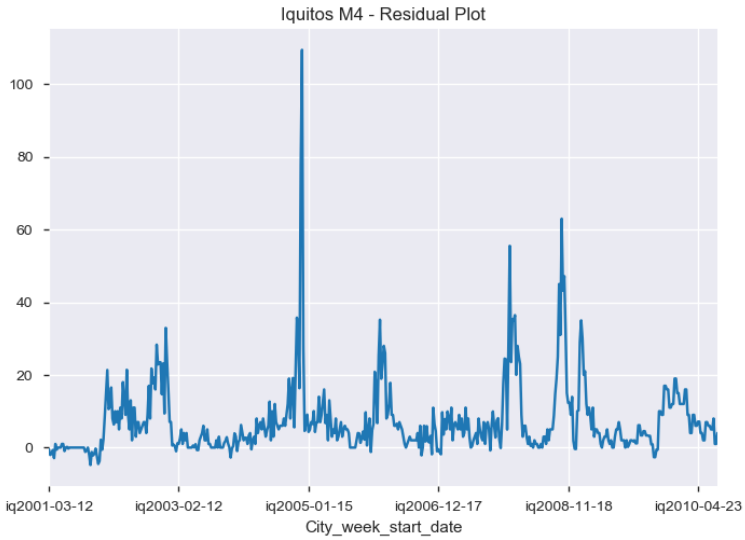


Figure 4.30: M4 – residual line plot

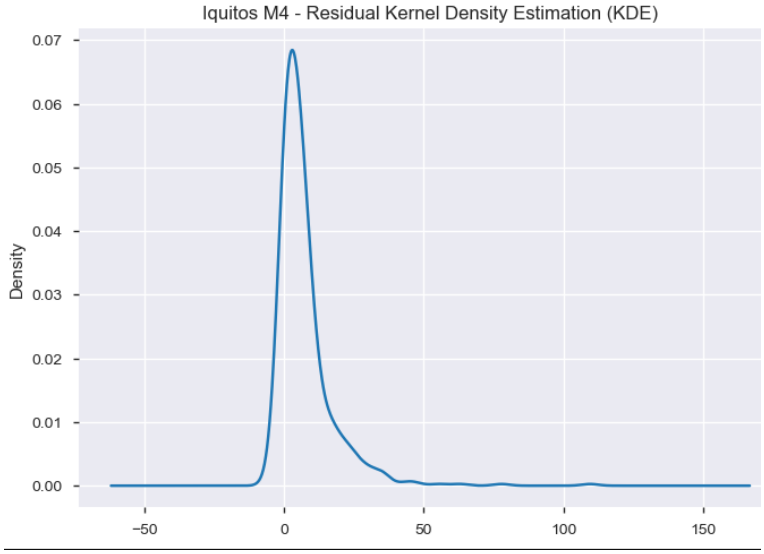


Figure 4.31: M3 – residual KDE

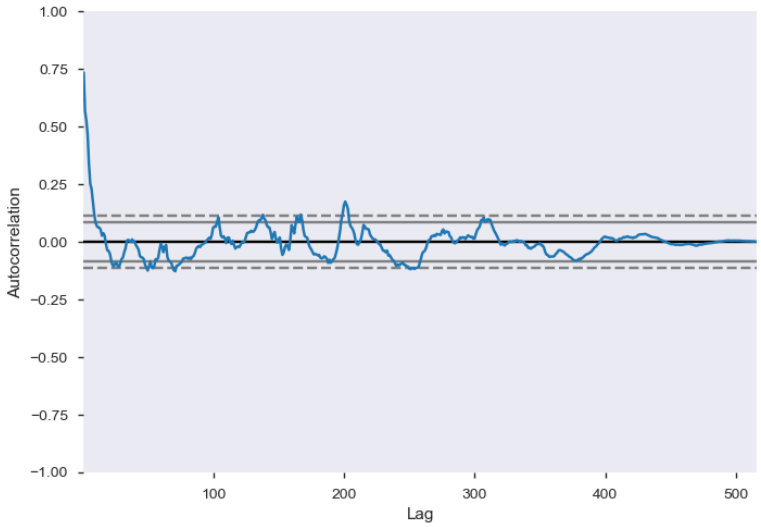


Figure 4.32: M4 – residual ACF plot

5. Discussion

5.1 Google Trend

The data extract for San Juan and Iquitos was not successful as both cities do not have ISO 3166-2 ticker that Google Trends API requires to identify region, state and city. San Juan is located in Puerto Rico which has only country level ISO ticker. While Iquitos has no ISO 3166-2 ticker but it is location within the Loreto region which carries "PER-LOR" as ISO regional ticker. Attempts to proxy Iquitos using regional level data from Loreto was unfruitful

- Data prior to 2004 cannot be extracted.

- Data for the remaining training period from 1-Jan-2004 until 1-Jul-2010 are all zeros for daily & weekly basis.
- Aggregation from daily to weekly and monthly did not tally, even taking the sampling effect into consideration.

5.2 Implausible values from MICE imputation

The single imputation of MICE produced a negative precipitation value which caused NaN for logarithmic transformation. In addition to the possibility of churning out implausible values, MICE based on stochastic regression face problems with heteroscedastic data. The new Predictive Mean Matching (PMM) Imputation overcome the two aforementioned weaknesses of stochastic regression based MICE (Statistical Programming, 2019).

5.3 Cross validation for time series analysis

The traditional K-fold Cross Validation (CV) assumed each observation is independent of the other. However, this is not the case with the temporal dependent (Cochrane, 2018) dengue cases and climatic data. Hence, the Time Series cross-validator (scikit-learn developers, 2018h) which is represented by Figure 5.1 was used in this research.

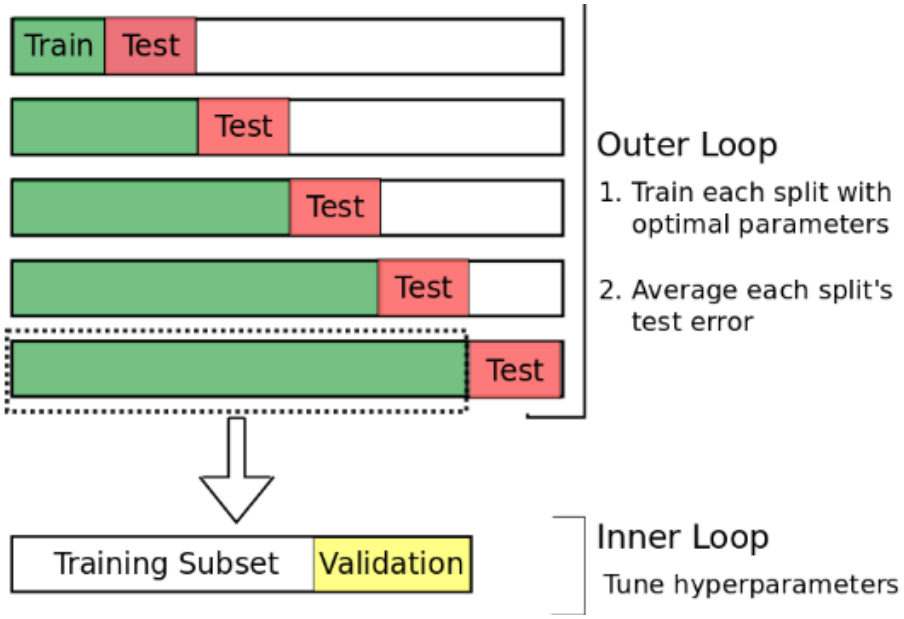


Figure 5.1: Nested cross validation (Cochrane, 2018)

A more sophisticated "evaluation on a rolling forecasting origin" in Figure 5.2 could also be adopted in the future.

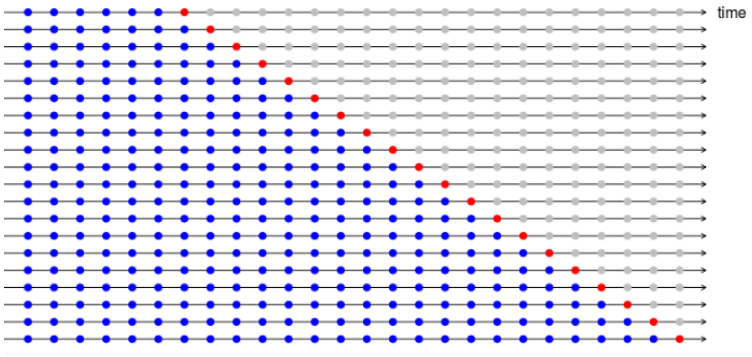


Figure 5.2: Evaluation on a rolling forecasting origin where the blue and red dots are training and test sets respectively (Hyndman, 2016)

5.4 Modelling for San Juan

Over-shrinkage (Musoro et al., 2016) occurred whereby the minimum MAE was found when optimal alpha (lambda in R's terminology) is larger than 1 in Figure 4.2 and 4.7 for M1 and M2 respectively. This could be caused by the lack of strong linear relationship whereby most of the weeks have very low dengue cases excepted during outbreak periods. Hence, the grid search were refined to work in the range whereby there are at least 5 sets of non-zero coefficient in the Lasso coefficient path in Figure 4.1 and 4.6 for M1 and M2 respectively.

Model M1 is favoured over M2 due to smaller overall MAE and better able to mimic the peaks. This means that the extensive engineered features in M2 did not add much value in predicting the dengue cases in San Juan, compared to the simpler M1. However, both M1-M2 did not capture the outbreaks based on time series plots (Figure 4.21, 4.24) and have significant cyclical components based on residual ACF plots (Figure 4.23, 4.26).

Table 5.1 – M1 vs. M2 for San Juan

Data set	Period	M1			M2		
		MAE	Alpha	Significant predictors	MAE	Alpha	Significant predictors
subtrain	10-Dec-1990 to 15-Jan-	1.3401	0.1	• station_max_temp_C • normal_reanalysis_tdtr_C –	1.6195	0.56234	• FE_normal_reanalysis_max_air_temp_C from 28-32 weeks ago.

	2003			• weekofyear • ndvi_nw • LN_renalaysis_precip_amt_kg_per_m2			• FE_station_diur_temp_rng_c from 23, 26-32 weeks ago. • FE_ndvi_nw of 28, 30-32 weeks ago. • FE_station_min_temp_c of 6 weeks ago. • FE_LN_precipitation_amt_mm of 32 weeks ago.
subtest	22-Jan-2003 to 25-Mar-2008	1.1104			1.1781		

5.5 Modelling for Iquitos

Over-shrinkage occurred for M4 model (Figure 4.17) whereby minimum MAE was found for alpha larger than one. However, both M3 (Figure 4.12) and M4 (Figure 4.17) have non-unique alphas whereby MAE charts do not have unique peak.

According to Tibshirani (2012), the non-unique Lasso solutions could occur when

- p > n with p = number of variables p and n = number of observations.
- Discrete predictors were used.
- Post-processing was carried out on continuous predictors.

Taylor (2012) asserted that if X^TX of formula (2) is invertible, the no unique Lasso solution can be found.

$$\hat{\beta} = (X^T X)^{-1} X^T Y \qquad (2)$$

p = 242 and n = 484 for Iquitos yields p / n = 0.5 the first condition is not fulfilled. weekofyear is the only discrete variable used in both cities but it was standardized, hence nullify the second condition. No further processing was none on the continuous variables in both cities, which invalidate the third condition. The remaining invertible possibility of Iquitos' X^TX needs to be further investigated and is outside the scope of this research.

Similar to San Juan's over-shrinkage problem in, Iquitos' grid search were refined to work in the range whereby there are at least 5 sets of non-zero coefficient in the Lasso coefficient path in Figure 4.11 and 4.16 for M3 and M4 respectively. Model M4 is favoured over M3 due to smaller MAE in subtest and better able to capture the cyclical component.

This means that the engineered features in M4 added value in predicting the dengue cases in Iquitos, compared to the simpler M3. However, both M3-M4 did not capture the outbreaks based on time series plots (Figure 4.21, 4.27).

Table 5.2 – M3 vs. M4 for Iquitos

Dataset	Period	M3			M4		
		MAE	Alpha	Significant predictors	MAE	Alpha	Significant predictors
subtrain	12-Feb-2001 to 6-Aug-2007	1.7303	0.1	• station_min_temp_c • weekofyear LN_station_precip_mm • ndvi_ne • LN_reanalysis_precip_amt_kg_per_m2 station_avg_temp_c	1.7741	0.3162	• FE_LN_reanalysis_precip_amt_kg_per_m2 from 13 weeks ago. • FFE_normal_reanalysis_min_air_temp_C from 13 weeks ago. • FE_station_avg_temp_c of 19-20 weeks ago. • FE_normal_reanalysis_tdtr_C from 11 weeks ago. • FE_station_min_temp_c of 0, 2, 23 weeks ago. • FE_LN_precipitation_amt_mm of 32 weeks ago.
subtest	13-Aug-2007 to 28-May-2010	2.4149			1.1781		

5.6 Comparison of San Juan vs. Iquitos

Temperature is the key and consistent predictor in both cities, where the hotter the temperature, the more dengue cases.

However, the different effects were noted for

- The gap between the lowest and highest temperature of the day, whereby normal_reanalysis_tdtr_C has negative coefficient in San Juan but FE_normal_reanalysis_tdtr_C has positive coefficients.
- Precipitation whereby LN_renalaysis_precip_amt_kg_per_m2 has positive coefficient in San Juan but Iquitos' FE_LN_reanalysis_precip_amt_kg_per_m2 from 13 weeks ago and FE_LN_precipitation_amt_mm of 32 weeks ago have negative coefficients.

San Juan is more sensitive to year end and vegetation effects compared to Iquitos. On the other hand, the lagged climatic variables effects are more profound in Iquitos. The lengths of significant climatic lags are found to be consistent with those of Sougata, Acebedo and Chua (2017).

Table 5.3 – The significant predictors of San Juan vs. Iquitos

San Juan – M1	Iquitos – M4
• station_max_temp_C - the hotter, the more cases. • normal_reanalysis_tdtr_C - the bigger the gap between maximum & minimum temperatures, the lesser cases. • weekofyear - cases increase toward year end. • ndvi_nw - the more vegetation on the north-west direction, the more cases. • LN_renalaysis_precip_amt_kg_per_m2 - the more rainfall, the more cases	• FE_LN_reanalysis_precip_amt_kg_per_m2 from 13 weeks ago - the more rainfall, the lesser cases. • FFE_normal_reanalysis_min_air_temp_C from 13 weeks ago - the higher the minimum temperatures, the more cases. • FE_station_avg_temp_c of 19-20 weeks ago - the hotter the ground average temperature, the lesser cases. • FE_normal_reanalysis_tdtr_C from 11 weeks ago - the larger the temperature gap, the more cases. • FE_station_min_temp_c of 0, 2, 23 weeks ago - the higher the minimum ground temperature, the more cases. • FE_LN_precipitation_amt_mm of 32 weeks ago - the more rainfall, the lesser the cases.

5.7 Benchmarking against other nowcasting work

Two studies which nowforecasted dengue are

- Freeze, Erraguntla and Verma (2018) which forecast 1-week and 4-weeks ahead, based on the same DrivenData – DengAI competition.
- Shi et al. (2016) which forecast week 1 to week 12 ahead for Singapore.

Freeze, Erraguntla and Verma (2018) used non-linear terms of square and cube total_cases and weekly percentage change of total_cases which were found to be significant in nowcasting dengue. This research does not involve the derivation of total_cases as this information is not made available in the test dataset. The authors combined both cities in the same MLR, SVM, RF and Boosting models while this research created separate Lasso models for each city. The significant "region" indicator in Freeze, Erraguntla and Verma's model support the separate modelling of two cities.

Shi et al. (2016) built an ensemble of 12 Lasso sub-models instead of a single Lasso model. The authors applied logarithmic transformation of the total_cases in the form of log (1 + normalized case) which has the potential to reduce the right-skewed residuals faced in this research. The authors also included Breeding Percentage (BP), which measured the dominance of Ae. Aegypti among the breeding sites. This is a crucial predictor as different serotypes dominated dengue outbreaks at different times, as noted in Iquitos (The International Federation of Red Cross and Red Crescent Societies, 2011b). While NOOA provided the serotype breakdown in the 2015 competition, such information is only limited to the train dataset, hence unable to be utilized to forecast the test dataset.

This research along with Freeze, Erraguntla and Verma failed to predict the severe outbreak for Iquitos in 2010.

6. Conclusion

This is the first recorded attempt to use city level Google Trend data to forecast dengue. However, both San Juan and Iquitos do not have the corresponding ISO 3166-2 ticker to enable city-level data extraction. Feature engineering added value to Iquitos but not for San Juan nowcasting model. Temperature is the key and consistent predictor in both cities, where the hotter the temperature, the more dengue cases. San Juan is more sensitive to year end and vegetation effects while the lagged climatic variables effects are more profound in Iquitos. However, the different effects were noted for the two cities, namely the gap between the lowest and highest temperature of the day and precipitation.

Predictive Mean Matching (PMM) could be used to overcome the negative imputed values, followed by Bootstrapping and MI-LASSO (Musuro et al. 2014) to further validate the multiple imputations used in Lasso model. Current Lasso models could be complemented with other submodels in an ensemble framework to capture cyclical components of climatic variables.