Spatio-Temporal Models for forecasting food delivery demand

Peng Seng Ang
National University of Singapore
ang.peng.seng@u.nus.edu

Supervised by: Professor He Long

Many real industry problems involves spatio-temporal demand prediction, which requires predicting demand at a certain time across different locations. Most of the demand data are not continuous variables but are instead count variables, like the number of orders and number of transactions. Previous research studies have mainly implemented neural network methods for spatio-temporal problems. In this paper however, we would explore more interpretable methods and hence will focus on how we can apply classical time series models, such as Vector Autoregressive (VAR), Spatio-Temporal Autoregressive (STAR) models to exploit spatial correlations as well as also introduce a 3-step approach to improve forecast accuracy.

Key words: Spatio-temporal demand prediction, Time Series, Clustering, Vector Autoregressive models, Spatio-Temporal Autoregressive models

1. Introduction

This paper would explore the usage of classical statistical methods along with unsupervised learning methods, like clustering, for the purpose of spatio-temporal forecasting. Spatio-Temporal forecasting can be defined as forecasting a future value at a given location and a given time. The motivation for this paper comes from Liu et al. (2020), where the focus is to optimise assignments of lunch delivery orders to drivers in order to minimize the total delay of all drivers. In reality, demand is never deterministic and by having a more accurate approach to forecast the demand, the lunch delivery company could more effectively and efficiently assign orders to drivers to improve the overall delivery time.

The dataset used in both Liu et al. (2020) and this paper are from a food service provider in China that allows customer to place orders before a cutoff time in the day (e.g 10.00am) and the customer can expect to recieve their orders by a deadline (e.g anytime from 10.30am to 11.45am).

This problem is not only restricted to the abovementioned food provider. With the rise of e-commerce and the food ordering and delivery services, like GrabFood and FoodPanda, demand prediction and driver assignment problem would be an everyday concern for them too. Spatio-temporalal forecasting is also not limited to food delivery, and it has many other applications, like traffic prediction and weather prediction over time.

The focus of this paper would be to accurately model and forecast the demand at the different locations and time. Currently, most Autoregressive (AR) or Autoregressive Integrated Moving Average (ARIMA) models only consider temporal features when predicting demand. However, we believe including spatial features between the data points might improve forecast accuracy. In this paper, classical time series models, such as Generalized Linear Model (GLM), ARIMA, Vector Autoregressive (VAR) as well as Spatio-Temporal Autoregressive (STAR) models, that could include both spatial and temporal features would be explored. We also introduce a 3-step approach that combines unsupervised learning and classical statistical methods to improve forecast accuracy.

2. Literature Review

Spatio-temporal demand prediction has slowly gained popularity over the years with the rise of deep learning. There are various research on different deep learning methods used for spatio-temporal demand prediction. One example would be Wang et al. (2018), where they applied deep spatio-temporal convolutional Long Short-Term Memory (LSTM) methods for traffic demand prediction. In their paper, they used the passenger flow to build a spatial correlation matrix as well as the time taken to travel between stations to form a time cost matrix. The 2 matrices are then used as spatial features and would be combined with temporal time series features to pass through their LSTM network.

While most existing research regarding spatio-temporal forecasting makes use of deep learning methods, this paper would be focus only on using classical statistical method instead as the dataset we are using is small and classical methods are relatively more interpretable than deep learning neural networks. Another example of using classical time series methods for spatio-temporal forecasting would be Safikhani et al. (2017), where they proposed using generalized spatio-temporal autoregressive (STAR) model for predicting taxi demands across locations in New York City. Kurt and Tunay (2015) also provides the background and theoretical details of STAR models, which they used on a dataset of regional bank deposits. STAR models originated from the 1980s but has been relatively silent during that period due to lack of relevant datasets. We will attempt to use STAR models in this paper as explained in section 8.

Knight et al. (2016) describes and shows how they implemented a network autoregressive moving average model to model the number of cases of Mumps in UK counties. In their example, they also showed that they might achieve a better result by modelling the series separately as univariate time series, also suggested in Nunes et al. (2015) since the neighbouring counties does not provide a substantial amount of explanatory power. Other related literature includes Luna and Genton (2005) where they propose a model building strategy for spatially sparse but temporally rich data.

The dataset that we will be using in this paper have a unique feature in which our data points are count (integer) values. As such, classical models such as poisson regression that are used for count data would produce probabilistic result in the form of fractional numbers. Hence, classical statistical accuracy metrics, like Mean Squared Error, as well as proper scoring rules for probabilistic forecast should be considered to evaluate both the accuracy as well as the fitness of our predictions. Past studies related to the different metrics used to evaluate probabilistic forecasting was done. Czado et al. (2009) mentioned that probabilistic forecasting should aim to

maximise sharpness of predictive distributions subject to calibration, where sharpness is defined as the concentration of the predicted distribution while calibration refers to the statistical consistency between the forecasts and the actual observation. As such, some scoring rules that could be used would be the logaritmic score, quadratic score, spherical score and ranked probability score.

BigVAR and tscount are some libraries in R that would be used in this paper. Liboschik et al. (2017) provides the mathematical background and implementation of Generalised Linear Models (GLM) for count time series as a library (tscount) in R, and tscount also provides the calculation for the scoring rules for probabilistic forecasting. Nicholson et al. (2017) extensively descibes the background and implementation of the VAR models and BigVAR library for multi-variate time series.

3. Data

The data source used was an operational dataset from a food delivery service provider from Shanghai that includes delivery information for a 2-month period from 10 August 2015 to 30 September 2015 (excluding Saturdays) in 2015. The provider only provides delivery service for 90 minutes during lunchtime and the dataset has split the data into 15-minute time periods, and as such, each day would only consists of demand data for 6 time periods. Hence, our dataset has 839 locations with demand count data, in integer, for 204 time periods in total.

To include other exogenous variables, data from World Weather Online (https://www.worldweatheronline.com/shanghai-weather-history/shanghai/cn.aspx) was used to extract information on temperature, wind, gust, cloud, humidity, precipitation, pressure. One-Hot Encoding of weekdays was also performed for all the respective days and included as an exogenous variable.

The next section would conduct some exploratory analysis to understand more about the dataset as well as any potential challenges.

3.1. Exploratory Analysis

We would first do some exploratory analysis and check if there are any interesting relationships between the variables.

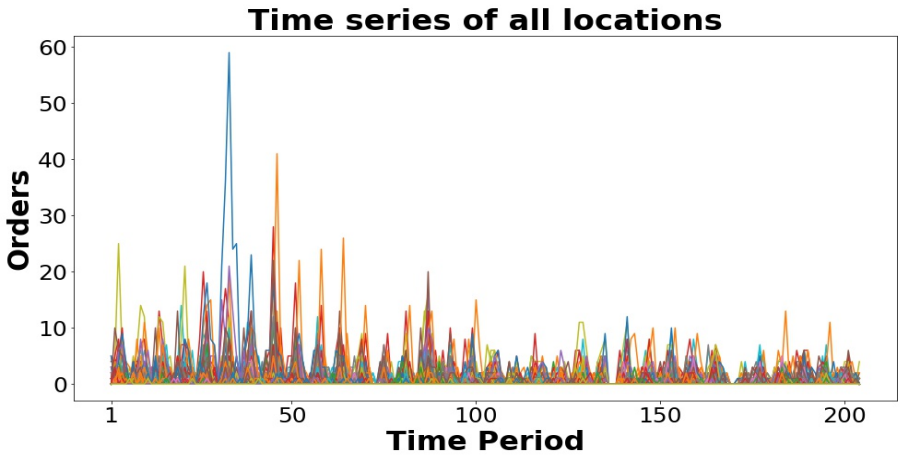

Figure 1 **Time series of all locations in the dataset. It can be observed that most locations have very low number of orders across the time period.**

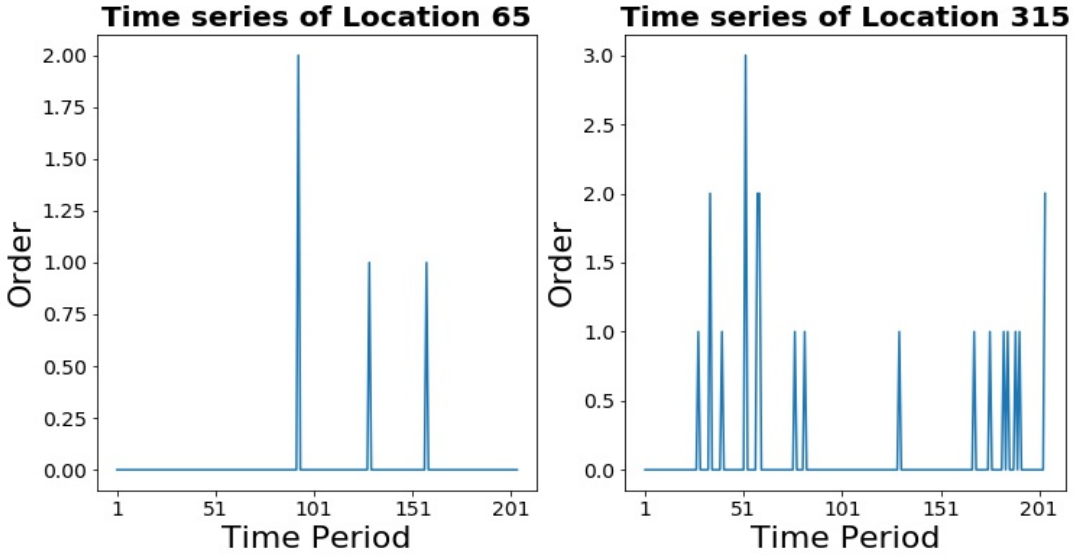

Figure 2 **Most locations have very sparse time series (left) while some have relatively more dense time series (right)**

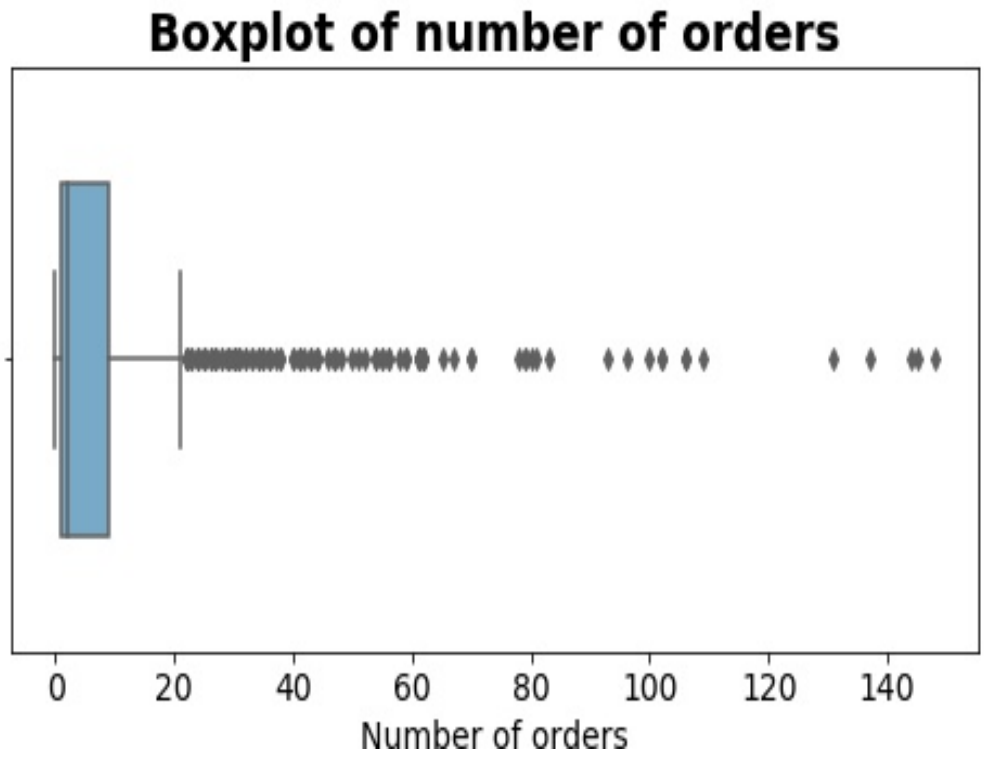

Figure 3 **Boxplot of counts. We can see that most of the locations have extremely low number of non-zero orders and further analysis showed that about 335 locations have just a maximum of one non-zero order throughout the 204 time periods.**

Next, we observe how the demand changes across locations over time.

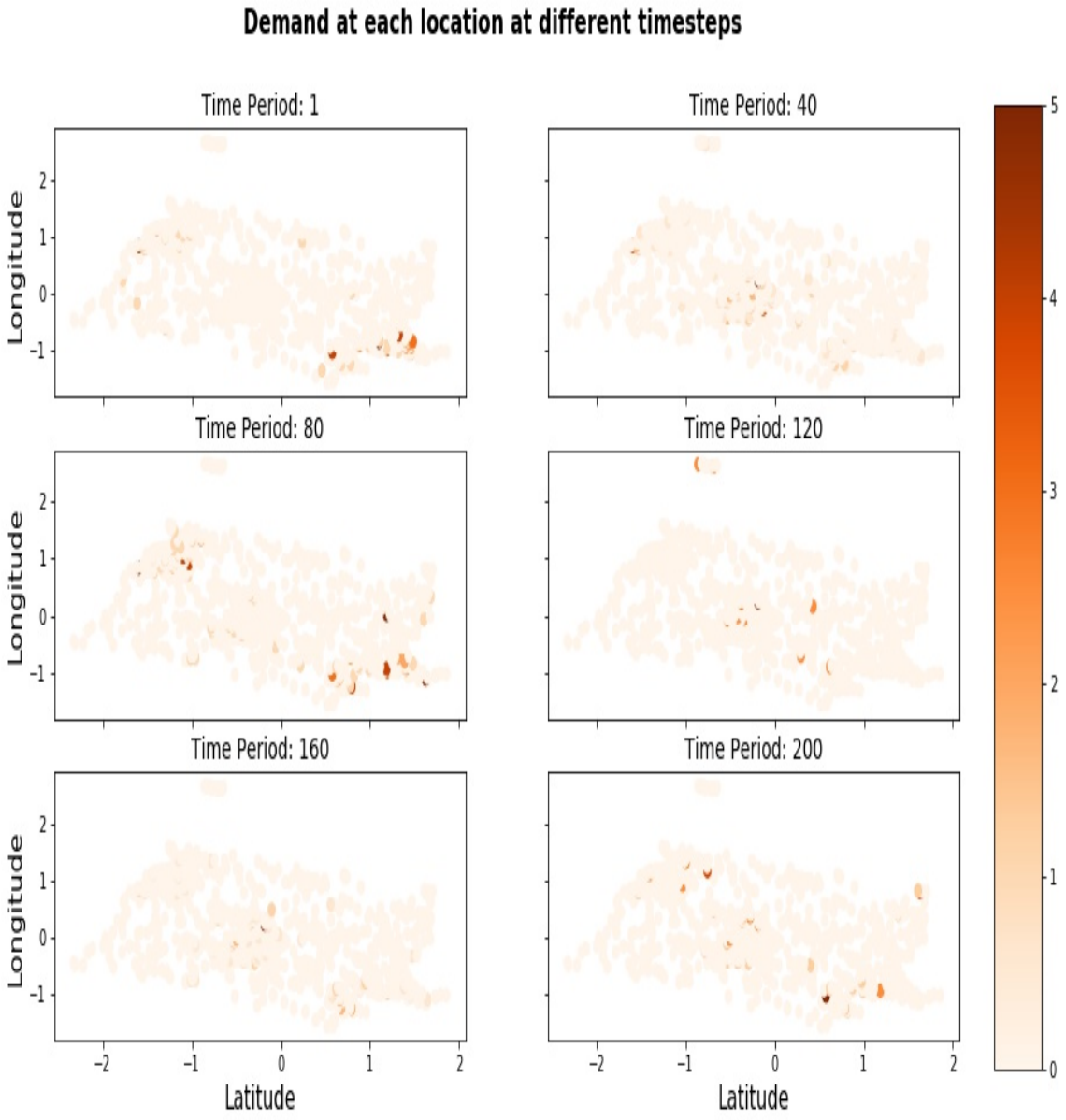

Figure 4 **Demand across locations over time. Plot suggests that the areas towards the right tend to have higher demand.**

Weather, temperature, days are also some external factors that might affect the demand. For example, perhaps a day with higher temperature might see more demand for food delivery since people might not be as willing to go out for food. These exogenous variables would be used in

our VARX model, which would be explained later in section 6.5. We performed some analysis on exogenous factors, such as the mean demand against temperature and against pressure.

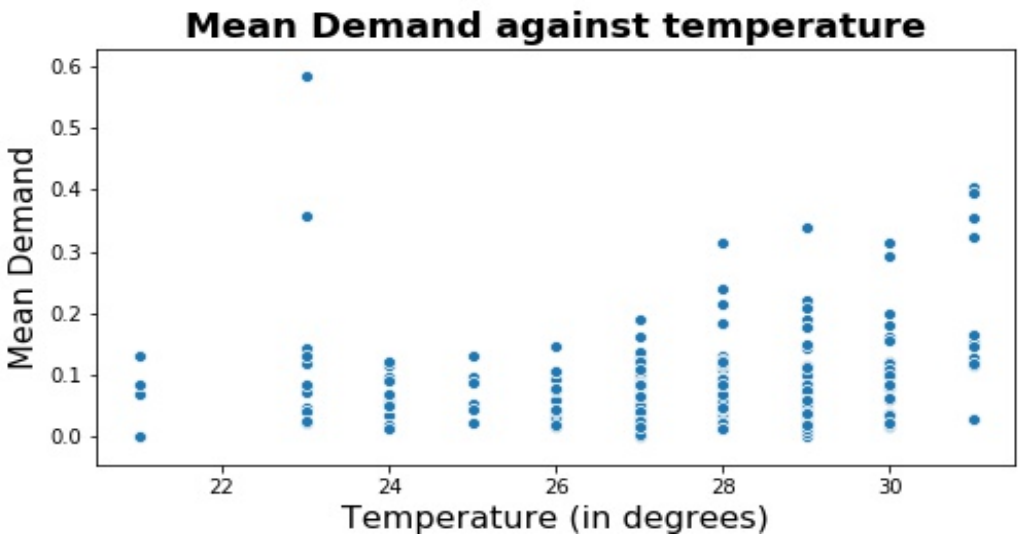

Figure 5 **Scatter plot of mean demand against temperature.**

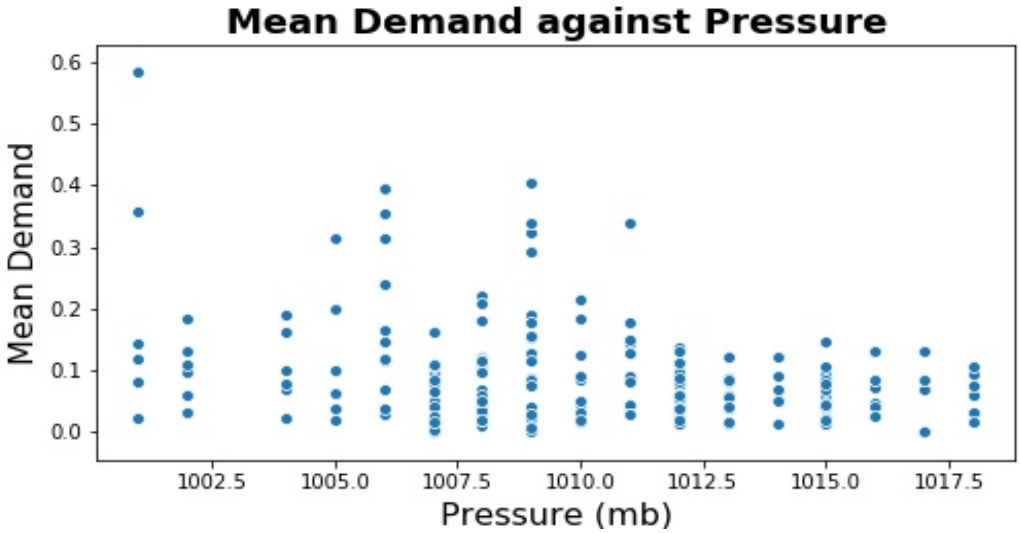

Figure 6 **Scatter plot of mean demand against pressure**

The scatter plot in Figure 4 visually display a slight positive relationship between temperature and mean demand across all locations whereas Figure 5 visually display a slight negative relationship between pressure and mean demand across all locations. We would explore usage and impact of the exogenous variables in predicting demand in the later portion of this paper.

3.2. Subsetting of data and train-test split

From Figure 1 in Section 3.1, the data is very sparse as there are many locations that have no demand counts for the majority of the time periods. Hence, to get a proof-of-concept model and a better idea of how our models perform, we first subset the dataset into a smaller one where we only consider locations with at least 50 non-zero counts across the time period, leaving us with 42 locations that meet this criteria. After the model is tested on this smaller dataset, we would evaluate the performance and then use the model to fit on the full dataset to evaluate its final performance.

To prevent overfitting of our model, the dataset was then split into training and test set by considering the first 33 days as the training set and the next 1 day as the test set. A period of 1 day was chosen as the test set duration because we are making a reasonable assumption that this model would only have to be run at the end of a working day to predict for the next working day, hence only required to predict one-day ahead demand.

Our training set would then have 198 time-step demand data for each location and test set would have 6 time-step demand data for each location.

The following sections of the paper is structured as follows. We would first discuss the different evaluation metrics that we would be using in section 4. A simple baseline ARIMA model would then be implemented in section 5. Section 6 and 7 follow a similar idea to the baseline model, but instead of fitting an ARIMA model on all locations, we fit a Generalized Linear Model (GLM) in Section 6 instead and in section 7, we fit a Vector Autoregressive Model using all the locations as variables. In section 8, we would implement a Spatio-Temporal Autoregressive (STAR) model and evaluate if it performs better than the VAR model. Section 9 would then introduce a 3 step modelling approach which would attempt to solve the problems and challenges we faced with the models so far as well as also performing the best out of all the models.

3.3. Challenges with dataset

There are 3 key challenges with our dataset:

1. Sparse data as most of the demand values are 0
2. Count data as all our values are in integer
3. High-dimensional, where we have time series for 839 locations

We will explore different methods to tackle the challenges mentioned in the following sections.

4. Evaluation Metrics

To evaluate time series models, the traditional way would be to use Root Mean Squared Error (RMSE) or Mean Squared Error (MSE) to monitor the predictive performance. However, since our dataset contains only count data and most of the models used in this paper are producing probabilistic forecast, in fractional numbers, we should also consider other metrics used for evaluation of probabilistic forecasting for count data, as suggested in Czado et al. (2009).

4.1. Mean Squared Error (MSE)

The classical metric that would be used for comparison would be Mean Squared Error (MSE), which is calculated by:

$$MSE = \frac{1}{n}\sum_{t=1}^{n} \|\hat{y}_t - y_t\|_2^2$$

where n is the number of data points, $\hat{y}_t$ is the predicted value at time t and y_t is the actual value at time t. Our model should aim to minimize the MSE in order to obtain a more accurate fit.

4.2. Quadratic Score

Calibration refers to the statistical consistency between forecasts and actual observations while sharpness represents the concentration of the predicted distribution and as mentioned in Czado et al. (2009), proper scoring rules would ensure that we address both calibration and sharpness of the forecast while the classical metric MSE do not. Hence, in addition to the classical MSE metric, we will also use a proper scoring rule, the quadratic score (qs), to evaluate our predictions. The usage of quadratic score to evaluate time series models for counts was also suggested by Wecker (1989). Quadratic score can be defined as:

$$qs(P, x) = -2p_x + \|p\|^2$$

where P represents the predicted distribution, x represents the actual observed value, p_x represents the probability mass at the observed count x and we can compute $\|p\|^2 = \sum_{k=0}^{\infty} p_k^2$ using methods explained in Appendix A of Czado et al. (2009). Quadratic score is also a proper scoring rule as it follows the property:

$$qs(x, x) \leq qs(P, x)$$

In this paper, we will calculate and report the quadratic score (QS) of our models by this formula:

$$QS = \frac{1}{n}\sum_{t=1}^{n} qs(P_t, x_t)$$

where n represents the number of data points. We should aim to minimize the quadratic score to provide a sharper fit.

5. Baseline Model

In this section, we would build a simple baseline model by just fitting an ARIMA model on all the locations individually. In the later sections of the paper, we would try other approaches and compare the results against the baseline model.

5.1. ARIMA models

Autoregressive Integrated Moving Average (ARIMA) models are one of the most commonly used models for time series (Farhath et al. (2016)). ARIMA models consists of 3 processes, mainly the Autoregressive (AR) process, the Integrated (I) process and the Moving Average (MA) process (Fattah et al. (2018)). The AR process assumes that every observation can be expressed as a linear combination of its past values and a AR(x) process would mean using x number of lagged values. The MA process assumes that each observation can be expressed as a linear combination of its current error term as well as its past error terms and a MA(x) process would mean using x number of past error observations. The Integrated Process states the number of times the time series could undergo differencing to ensure that the eventual time series is stationary.

Hence, an ARIMA model is usually represented by ARIMA(p,d,q), where p represents the number of autoregressive terms, d represents the number of differencing steps needed for stationarity, and q represents the number of lagged forecast errors.

The general equation of a ARIMA(p,d,q) is:

$$\hat{y_t} = \mu + \sum_{i=1}^{p} \phi_i * y_{t-i} - \sum_{j=1}^{q} \theta_j * e_{t-j}$$

where $\hat{y_t}$ refer to the predicted value at time t, μ refer to a constant, while $\sum_{i=1}^{p} \phi_i * y_{t-i}$ refers to the AR terms and $\sum_{j=1}^{q} \theta_j * e_{t-j}$ refers to the MA terms.

5.2. Baseline ARIMA Result

As a baseline model, each of the locations was assessed individually and a suitable ARIMA model was built for each location. Auto-arima function from R was used to implement this as it would automatically determine if the series is stationary, perform differencing if required, as well as finding the optimal parameters for p,d and q.

The out-of-sample MSE for this baseline model on the smaller dataset is **46.87** and the quadratic score is **-19.12** while the MSE for dataset of all locations is **71.73** and the quadratic score is **-772.45**. A forecast plot on a random location is shown below:

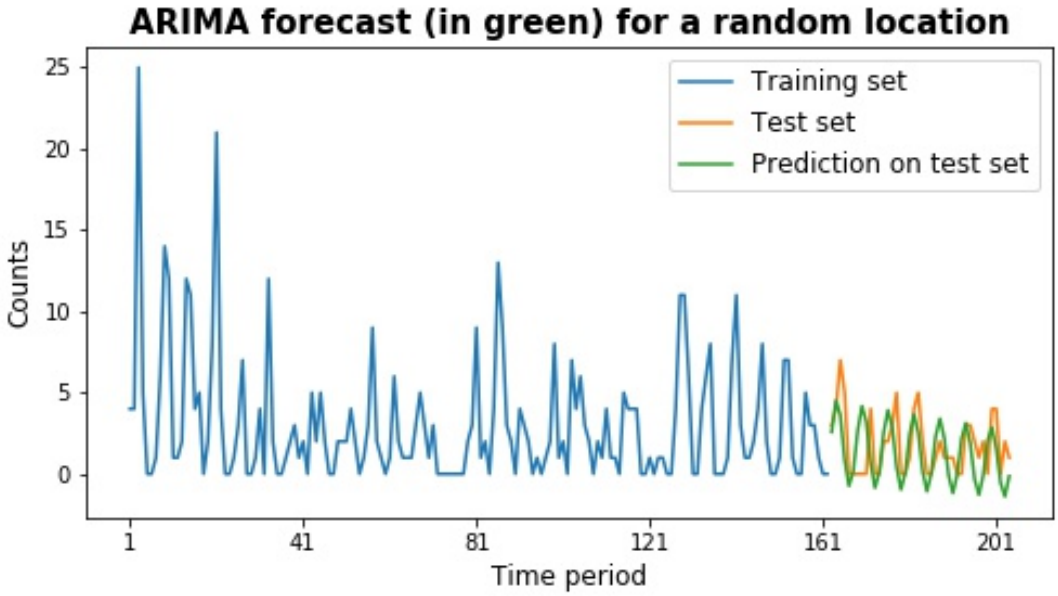

Figure 7 **ARIMA forecast on a random location**

The next section would explore using the Generalized Linear Model (GLM) approach.

6. GLM Model

The dataset that we are using follows a count time series, which means the observations are non-negative integers. A flexible and commonly used model for count time series is the Generalized Linear Model (GLM) Nelder and Wedderburn (1972). GLM model makes an assumption that each value is conditionally dependent on its past values. According to Liboschik et al. (2017), an advantage of GLM models over typical ARIMA models is that GLM can describe covariate effects and negative correlations in a more direct way. GLM normally take the form of:

$$g(\lambda_t) = \eta^T X_t$$

where g represents a link function, and $\lambda_t = E(Y_t|F_{t-1})$, where F_{t-1} represents the historical values up to time t, which is the conditional mean of the time series . η represents a parameter vector that corresponds to the covariates and X_t represents the a vector of the values of the previous time steps in the time series.

6.1. Model Implementation

As mentioned in section 3.2, we would first assess our model on the smaller dataset before testing it on the full dataset. Using the R package tscount from Liboschik et al. (2017), we fit each of the locations individually by a GLM model. For the parameters, we used the past **6** lagged values as well as the **identity function** as the link function, with **poisson distribution** as the conditional distribution. The out-of-sample MSE for this baseline model on the smaller dataset is **38.16**, which is better than just using ARIMA on the same dataset. The quadratic score for GLM on the smaller dataset is **-19.90**.

The output for GLM model is in a fractional number instead of an integer. The reason is because the output of the GLM model, using poisson distribution, gives the expected value which might not be an integer. Hence, to convert our predictions to an integer for practical purposes, we would round the predictions to the nearest integer since the nearest integer would be the value that is predicted to occur most often in the distribution.

However, when it is implemented on the full dataset, the MSE of the GLM increases to **82.42**, which performs worse than just applying ARIMA on every location, which gives a MSE of 73.29. The quadratic score for GLM on the full dataset is **-656.94**, which also lose out to the quadratic score of using ARIMA model.

6.2. Model Diagnostics

To validate and verify if our fitted model is adequate, model checking would be performed by performing the following residual analysis.

6.2.1. Residuals plots

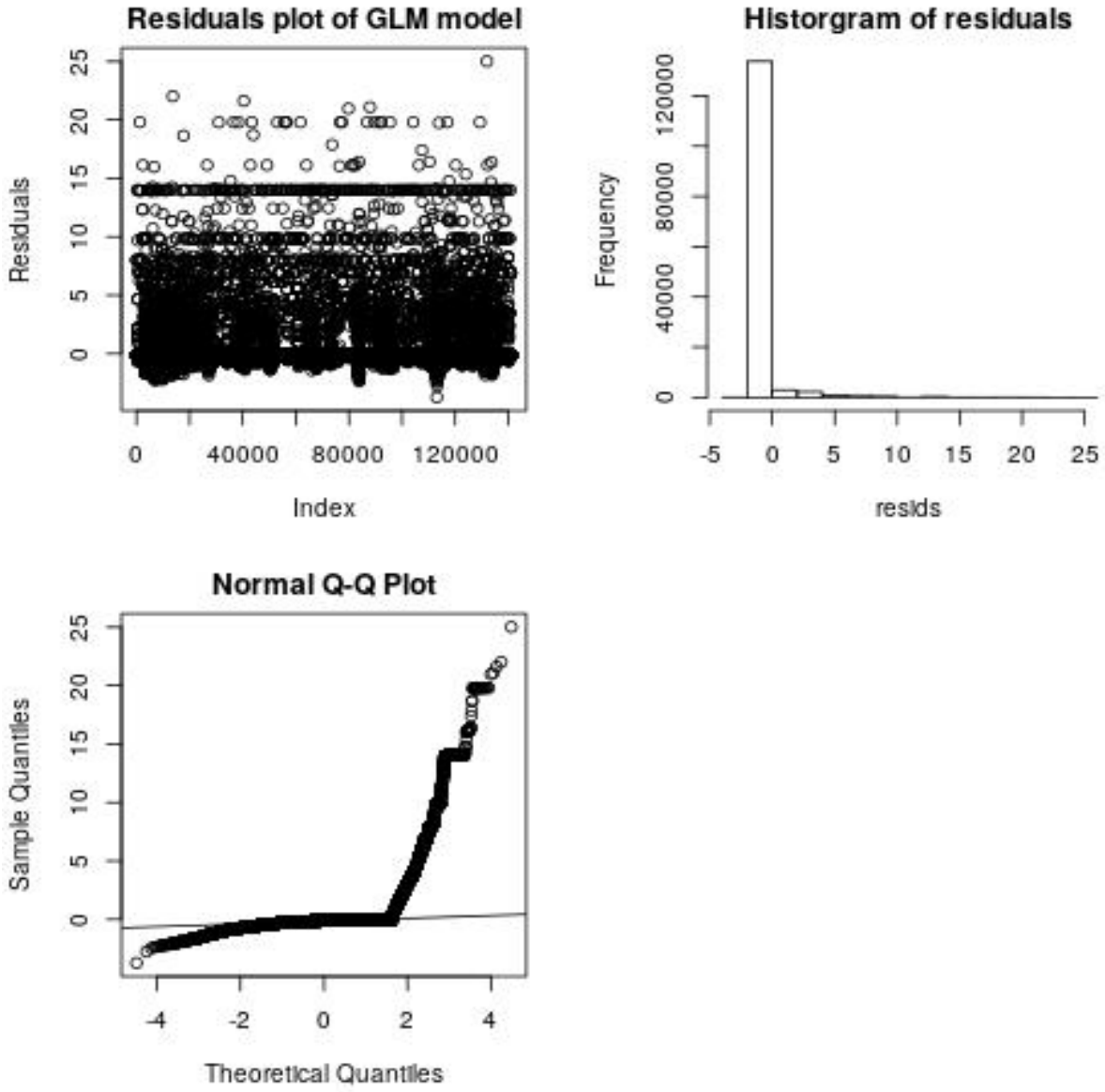

Figure 8 **Residuals for GLM Model**

The diagnostic plots in Figure 8 shows that while most residuals are randomly scattered around -5 to 10, there are still many points above 10, and this also imply that the GLM model produces residuals that does not follow the normal distribution well. Although this is not ideal, this might be attributed to the demand data following a poisson distribution instead of a normal distribution.

6.2.2. Residuals against Predicted values

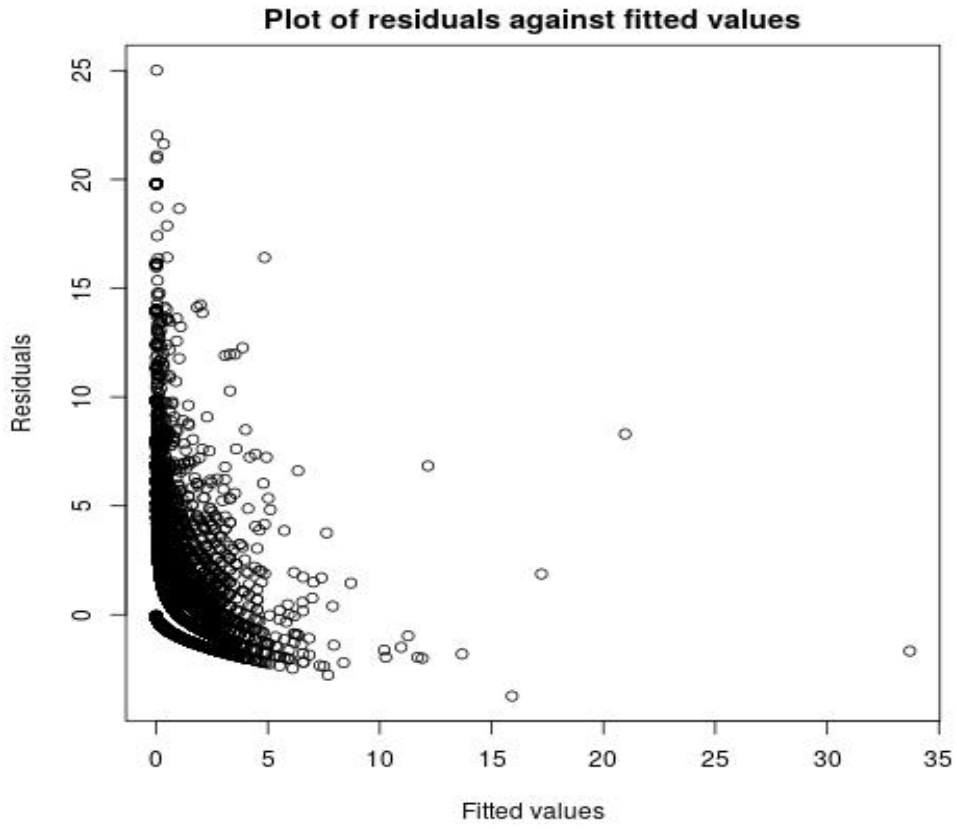

Figure 9 Residuals against Predicted values for GLM Model

The above plot of residuals against the predicted values suggests heteroscedasticity between residuals, or non-constant variance between the residuals as the predicted value increases. This heteroscadasticity among residuals would sometimes occur in a Poisson GLM, as mentioned in Molenaar and Bolsinova (2017).

6.3. Model Limitation

Similar to the baseline ARIMA model, it would be a relatively expensive and time-consuming process as every location has to be individually fitted to a GLM model. Also, each location model only uses its past values and does not take into account data from the other locations. The following sections explore other types of model which would use data from other locations as predictors.

7. Vector Autoregressive (VAR) Model

7.1. Details of VAR Model

Vector Autoregressive (VAR) models are one of the most commonly used model for multivariate time series, particularly in fields like economics and financial time series as shown in Bjørnland (2014). VAR models are very similar to multivariate linear regression models and hence, methods used to perform inferencing on linear regression models can also be applied to VAR models. VAR(p) represents a VAR model of order p and can be written as:

$$y_t = v + \sum_{i=1}^{p} \phi_i y_{t-i} + \alpha_t$$

where y_t represents the predicted value at time t, p is the number of lagged endogenous variables used, y_t is the value at time t, v is a constant vector, ϕ_i are coefficient matrices for $i > 0$ and α_t are independent and identically distributed random vectors.

7.2. Motivation

As with many spatio-temporal demand prediction, spatial features could be an important feature if there exists spatial correlation. To check if there exist spatial correlation between the locations, a correlation heatmap is plotted. For clear illustration purposes, instead of showing the matrix of correlation for all 839 locations, only the 42 locations with at least 50 non-zero counts would be shown.

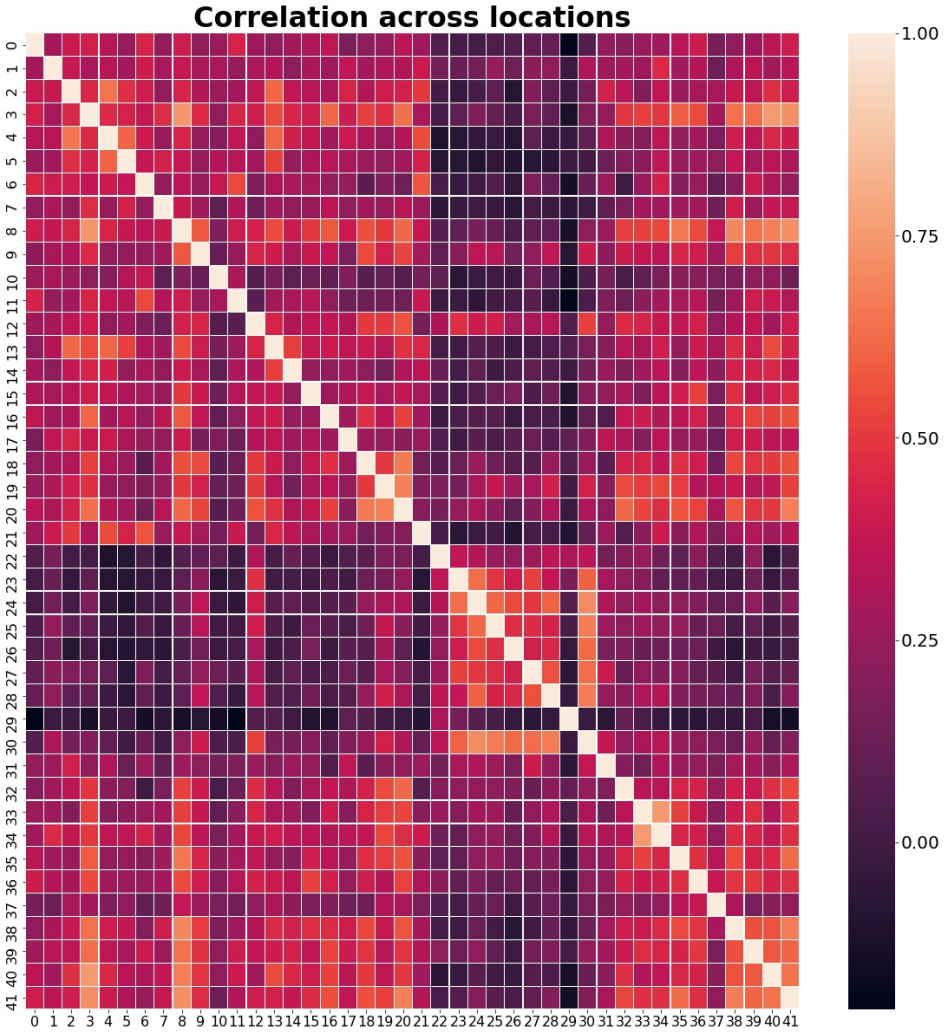

Figure 10 **Spatial Correlation Heatmap on the locations with at least 50 non-zero counts. Axes represent the 42 locations**

We can observe that there are certain locations' time series which has relatively high correlation with other locations' time series. This thus provides the motivation to use the existence of spatial correlation and explore the usage of VAR model.

7.3. Stationarity Condition

For a univariate time series, it is important for the time series to be transformed into a stationary series and Augmented Dickey-Fuller (ADF) test can be used to perform unit root test for stationarity, as shown in Xiao and Phillips (1998) and Mushtaq (2011). For a multi-variate time series, if the series are unit-root non-stationary, regression models might show a statistially significant, but false, coefficients and results simply because the series are coincidentally increasing over time, hence applying the VAR model on non-stationary series could lead to spurious regression, also shown in Baumöhl and Lyócsa (2009).

7.4. Implementation of VAR Model for our dataset

To test for stationarity, we would be performing the standard ADF test to test for stationarity. The result of the ADF test showed that time series of certain locations are not stationary. We then take the first difference of each series and after differencing, ADF test was performed on each series the time series of each location have been checked and are stationary. It is noted that while it is possible to perform differencing on every series, it might cause over-differencing, leading to inaccurate results, as mentioned in Tsay (2013).

After we applied a differencing order of 1 to the dataset, BigVAR Library in R was used to fit a VAR model with a selected p of 12, representing the usage of 12 lagged values, using time series of every location as predictors. The results for the VAR model is discussed in the later subsection.

7.4.1. Cointegration

Box and Tiao (1977) shows that it is possible to linearly combine various unit-root nonstationary time series to form a stationary series. The term Cointegration, first mentioned in Engle and Granger (1987), states that although some or all the time series might be unit-root nonstationary individually, these time series can be said to be cointegrated if there exists a possible linear combination of them that would form a stationary series. Intuitively, 2 series are cointegrated if they move together and the distance between them remain stable over time.

7.4.2. Johansen Test for Cointegration

While Cointegrated Augmented Dickey Fuller Test, commonly used for Pairs Trading, can be used, it is able to be applied on only 2 separate series. The next best approach is the cointegrating tests for VAR model, called the Johansen's Cointegration Test. However, one limitation is that it can only be used to check for cointegration between a maximum of 12 variables. For further elaboration on the Johansen's Cointegration Test, please refer to Johansen (1991). If there exists cointegration between variables, a Vector Error Correction Model can be formulated.

However, since our dataset has 839 variables (locations), we are unable to apply the cointegration test or accurately calculate the significant values of more than 12 variables and hence unable to determine correctly the number of cointegration vectors needed.

7.5. VARX Model

VAR models can also be extended to include exogenous variables. A VARX(p,s) (with exogenous variables) model can be expressed as:

$$y_t = v + \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{j=1}^{s} \beta_j x_{t-j} + \alpha_t$$

where p is the number of past endogenous variables used, s is the number of past exogenous variables used, y_t is the value at time t, v is a constant vector, ϕ_i are coefficient matrices for endogenous coefficient matrix for $i > 0$, β_j are coefficient matrices for exogenous coefficient matrix for $j > 0$, x_{t-j} represents the value of the exogenous variables at time t - j, where j is the lagged period for exogenous variables and α_t are independent and identically distributed random vectors.

7.6. Implementation of VARX Model on our dataset

Our dataset uses additional exogenous variables like temperature, wind, gust, cloud, humidity, precipitation, pressure as well as one-hot encoding of the day of the week. Our dataset now would have 839 endogenous variables/locations and 13 exogenous variables. Based on performance results and taking computational power limitation into consideration, the optimal number of lag (p) that was chosen for endogenous variables is 6 and number of lag for exogenous variables is 1.

Similar to VAR above, BigVAR library was used to fit a VARX model taking in the 839 locations' time series as endogenous variables and the 13 exogenous variables. The model was then use to compute predictions for the 6 periods ahead and would be compared with the test set.

7.7. Model Checking

The following residual analysis are then performed:

7.7.1. Whiteness of Residuals

To ensure our fitted model is adequate, the resiudals should behave like a white noise series. The plots below shows the distribution of our residuals for the VAR model and the VARX model.

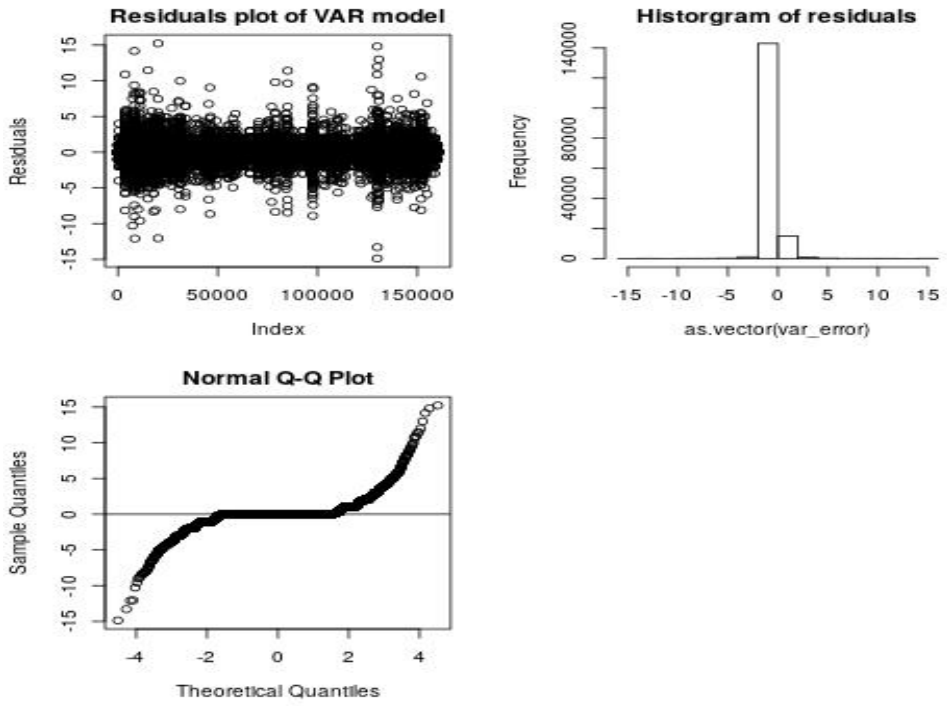

Figure 11 **Residuals for VAR Model**

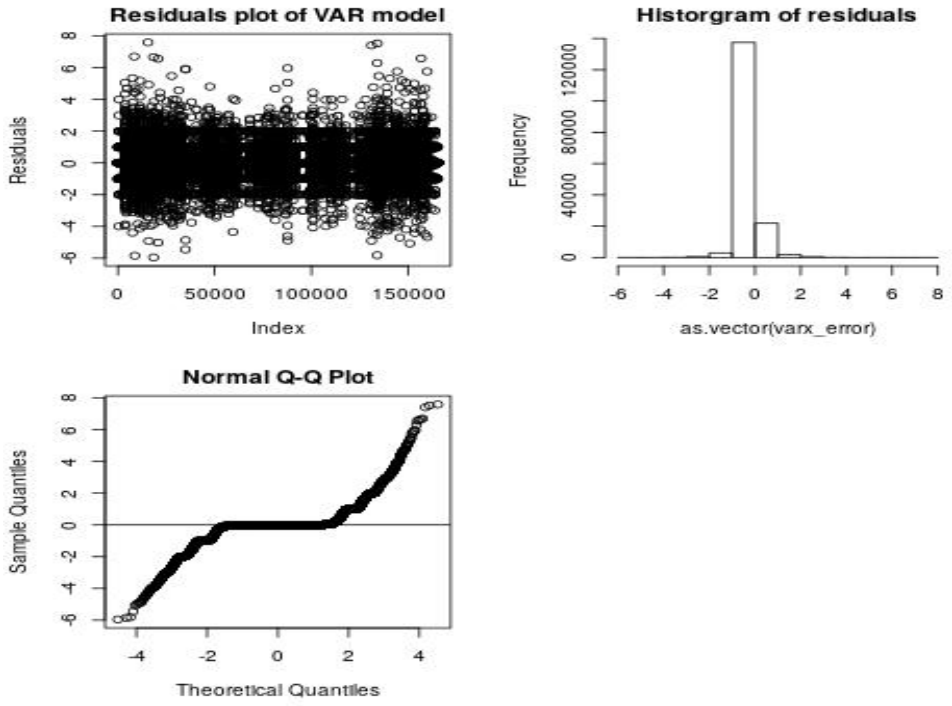

Figure 12 Residuals for VARX Model

From Figure 11 and 12, we can see that the residuals are mostly randomly scattered and they roughly follow a normal distribution, although it performs rather badly on the lower end and higher end of the outliers.

7.8. Results

On the small dataset, the MSE and quadratic score of the VAR model is **47.28** and **-15.11** respectively while on the full dataset, the MSE and quadratic score using VAR model are **82.87** and **-750.94** respectively. When fitting VARX model on the smaller dataset, the MSE and quadratic score are **45.76** and **-16.23** while on the bigger dataset, the MSE and quadratic score are **73.69** and **-727.02**. Since the VARX includes exogenous variables and have quite a significant improvement in result as compared to the VAR model, it shows that the exogenous variables used do have a moderate amount of explanatory power.

7.9. Model results and limitations

We also note that the output for VAR or VARX model is in a fractional number instead of an integer. To convert our predictions to an integer for practical purposes, we round the predictions to the nearest integer. As the VAR and VARX models do not perform as well as the baseline ARIMA models, the next section would explore using STAR models as another alternative model that can include spatial features.

8. Spatio-Temporal Autoregressive Model (STAR)

Safikhani et al. (2017) has evaluated ARMA, VAR and Spatio-Temporal Autoregressive (STAR) model to predict taxi demands in New York City and have found that STAR models performs much better in his paper. This section would explore the use of STAR models on our dataset.

8.1. Details of STAR models

We will outline the basic theoretical framework for STAR models. STAR model is a time series model that attempts to use the connections between variables across both time and space domains. For spatio-temporal data, it is common to have interdependencies between geographical locations and taking into account these spatial interdependencies and relationships might lead to a more accurate model. The main problem, however, is how to accurately define these spatial relationships. Spatial relationships between the locations can vary at different time periods and hence could have different spatial weights across time.

We will also introduce the concept of spatial lags. Temporal lag is relatively more straightforward to define as it directly shifts the observed value to a different time in the past. However, defining spatial lags are not as direct. The first step would be to determine and group the locations into neighbourhood sets before defining the first, second and any higher-order neighbourhood sets for each location.

After we have defined our neighbourhood set for each location, we can calculate $(w_{ij}^{(s)})$ which represents the spatial weight between location i and j at s^{th} order spatial lag. Spatial weights should follow these properties:

$$w_{ii}^{(s)} = 0$$
$$w_{ij}^{(s)} \geq 0$$
$$\sum_{j} w_{ij}^{(s)} = 1$$

The properties state that the sum of the weights for all neighbouring location j for each location i should sum up to 1 and have non-zero values for location i's first order neighbours. We can then combine all regions' weights into $W^{(s)}$, which represents a spatial weight matrix of N x N where N is the number of locations that our dataset has. The matrix would be make up of zeros for its diagonal elements while the rest will be filled with positive numbers. One way to calculate the weights would be to uniformly distribute the weights for every neighbouring location j of location i.

STAR models of the form STAR(p,k) can then be expressed as:

$$y_t = \sum_{l=1}^{p} \sum_{s=0}^{k_l} \phi_{ls} W^{(s)} y_{t-l} + \epsilon_t$$

where p represents time lag and k reprsents spatial lag. y_t represents the predicted demand at time t, ϕ_{ls} represents diagonal matrices, ϵ_t represents white noise with mean vector 0. More details regarding STAR models can be found in Kurt and Tunay (2015).

8.2. Implementation of STAR model for our dataset

In this paper, we assume that spatial relationships remain constant throughout the time periods and we leave computing spatial relationships that differ over time to be explored in future research. We also assume that for all locations, their first-order neighbourhood set would include all other locations. For this paper, we only take into account the first-order neighbourhood. To calculate the weight matrix, we uniformly distribute the weights for all neighbouring locations of each location.

The package ′gstar′ from R was used to implement the STAR model. STAR(5,1) was used for the smaller dataset while STAR(1,1) was used on the full dataset.

8.3. Results of STAR model

On the small dataset, the MSE of the STAR model is **39.76** and the quadratic score is **-20.27**. On the full dataset, the MSE of the STAR model is **81.50** and the quadratic score is **-656.32**. The MSE of the STAR model is slightly better tha the MSE using VAR model on high-dimensional data, which is similar to what Safikhani et al. (2017) observed too. However, the MSE of STAR model is still higher as compared to using the VARX model.

While VAR, VARX and STAR models are an improvement from the GLM model and also beats the ARIMA baseline model on the smaller dataset, the 3 models fail to beat the baseline model MSE of 71.73. This might be because in the baseline model, since each location is fitted with an individual ARIMA model, those sparse locations would have predictions of all 0, which would be deemed to be more accurate.

In the smaller dataset, since we only used locations that are not sparse (locations with at least 50 non-zero counts), and the VAR/VARX/STAR model uses all locations' demand values, the existence of spatial correlation would cause the VAR/VARX/STAR model to perform better than the ARIMA. However, when using VAR/VARX/STAR models on the full dataset, which contains locations with sparse demand, they would not beat the ARIMA model because the presence

of non-zero coefficients would cause many predictions to be non-zero, hence producing a more inaccurate prediction as compared to the baseline ARIMA model.

Having noticed the abovementioned problem in locations with sparse demand. the next section would introduce a 3-step approach to fix these challenges.

9. 3-step Approach

In this section, we introduce an alternative 3-step approach. The overall steps are summarised here:

Step 1. Perform Clustering on the time series using one of the distance metric:

1.1 Euclidean distance between geo-location coordinates

1.2 Correlation between time series

1.3 Dynamic Time Warping (DTW) distance between time series

Step 2. Aggregate the clusters by performing a summation the respective locations' demand over time. Train a VAR or a STAR model on the clusters to predict the total demand for each cluster.

Step 3. Re-allocate the total demand of each cluster to each location in the cluster while maintaining a relative constant distribution as before.

By performing clustering and aggregation of the locations before reallocating the demand to each locations based on its past distribution, the predicted demand for each location would not purely be attained from past values of all individual locations. Instead, the clustering would reduce the total dimensionality of the data when being fitted to a VAR/STAR model to attain a more reasonable cluster demand prediction, before being reallocated to the locations based on its past distribution and this might provide a better prediction for locations with sparse demands. Similar methods and ideas can be found in Murray et al. (2015) and Lu and Chang (2014).

9.1. Clustering

A simple clustering was first done on the locations using their time series data in the training set. 3 different kind of distance metrics were used to perform clustering. The first metric used would be based on their geo-locations while the second and third metric used would be to cluster them based on similarity between their historical time series.

9.1.1. Using longitude/latitude as the distance metric

A reasonable assumption that could be made is that customers in the same neighbourhood would tend to have similar ordering behaviour due to similar residential status or similar working industry or culture. Hence, we would cluster the locations based on their geo-location coordinates (longitude and latitude). K-means clustering is performed using the euclidean distance of the locations as the distance metric.

9.1.2. Using Correlation Coefficient as the distance metric

We then also explored using correlation coefficient between the time series as the distance metric for K-means clustering. Since correlation measures the strength and direction of the tendency of any 2 time series to move together, we believe that by grouping locations with similar trends, our prediction model would be more reasonable and accurate. Correlation coefficient can be calculated by:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2(y_i - \overline{y})^2}}$$

where r_{xy} represents the correlation coefficient between time series x and y, x_i and y_i represents points in time i for time series x and y respectively and $\overline{x}$ and $\overline{y}$ represents the mean value for time series x and y.

9.1.3. Using Dynamic Time Warping as the distance metric

Finally, we explored using Dynamic Time Warping (DTW) Distance as the distance metric for Hierachical clustering.

DTW is one of the commonly-used algorithm for speech or audio recognition and it measures the similarity between time series by providing a elastic non-linear alignment between 2 time series. It is known for being effective for time series that have different length or speed. DTW is calculated by first creating a distance matrix and then finding the optimal warping path as shown in the 2 algorithms here:

Algorithm 1 Dynamic Time Warping (DTW) Algorithm to form distance matrix

1. Initialise a 2-dimensional matrix M, where the indices of the rows and indices of the columns represent each point in time series x and time series y.

2. Populate the matrix from bottom-left to top-right with each element $c_{i,j}$ of the matrix representing the distance between x_i, the i^{th} element of time series x and y_j, the j_{th} element of time series y, which is calculated by:

$$c_{i,j} = (x_i - y_j) + min(c_{i-1,j}, c_{i,j-1}, ci-1, j-1)$$

Algorithm 2 Using DTW Matrix to find optimal warping path

Require: distance matrix of dimension i*j obtained from DTW Algorithm

```
Let i = rows(matrix) and j = columns(matrix)
Let path = []
while (i != 1) and (j != 1) do
  if i==1 then
    j = j - 1
  else if j==1 then
    i = i - 1
  else
    if matrix[i-1,j] == min(matrix(i - 1, j), matrix(i, j - 1), matrix(i - 1, j - 1)) then
      i = i - 1
    else if matrix[i,j-1] == min(matrix(i - 1, j), matrix(i, j - 1), matrix(i - 1, j - 1)) then
      j = j - 1
    else
      i = i - 1, j = j - 1
    end ifpath.add((i,j))
  end if
end while
return path
```

We can also understand DTW visually by:

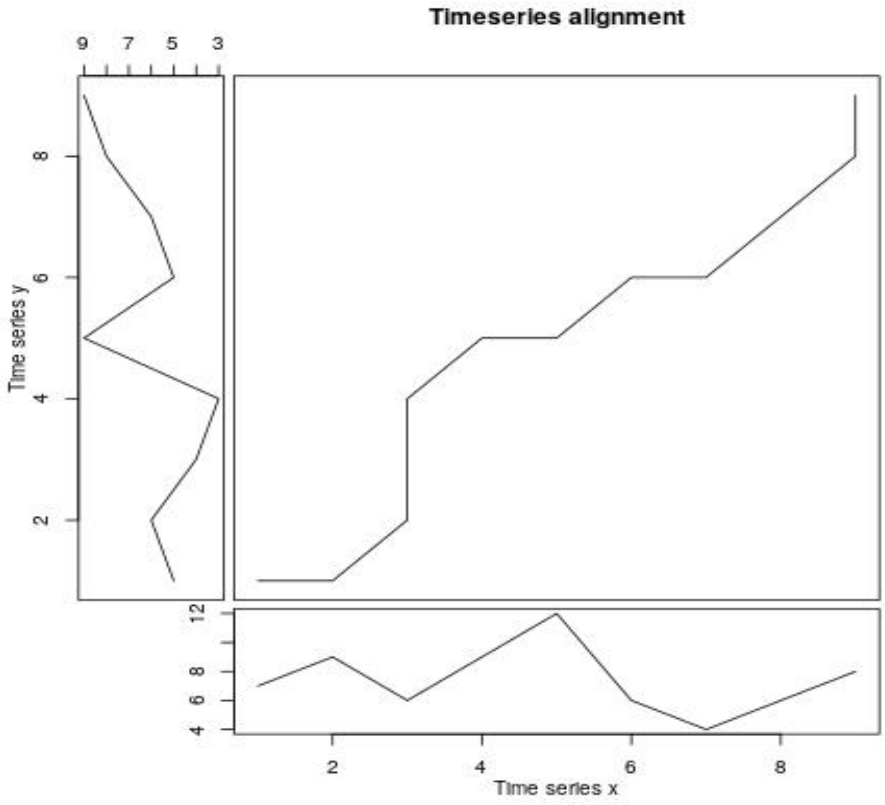

Figure 13 DTW Alignment Plot

The plot in the middle shows the optimal warping path between the 2 time series. The distance is calculated by the euclidean distance between the points of the 2 time series that lie along the warping path as shown in:

$$DTW\,Distance = \sqrt{\sum_{(i,j)\in WarpingPath}(x_i - y_j)^2}$$

As shown in Ding et al. (2008), DTW is also used commonly as a time series similarity measure for time series classification and it can perform as well as other state-of-the-art measures. More details on the applications and optimisation of DTW can also be found in Senin (2008)

9.2. Predicting total demand for each cluster

We first aggregate the locations of each cluster together such that:

$$\forall x \exists i \ni x \, \epsilon \, C(i)$$
$$D_{C(i)} = \sum_{l \epsilon C(i)} \sum_{j=1}^{n} x_{lj}$$

The first equation states that every location x will belong to one cluster. The second equation states that we obtain each cluster's time series by summing the past demand values at each time period of each location in the cluster. $D_{C(i)}$ represents the total demand (from training set) of the cluster C(i) and x_{ij} represents the training set demand value at location i at time j.

We then take a look at the correlation heatmap of the clusters.

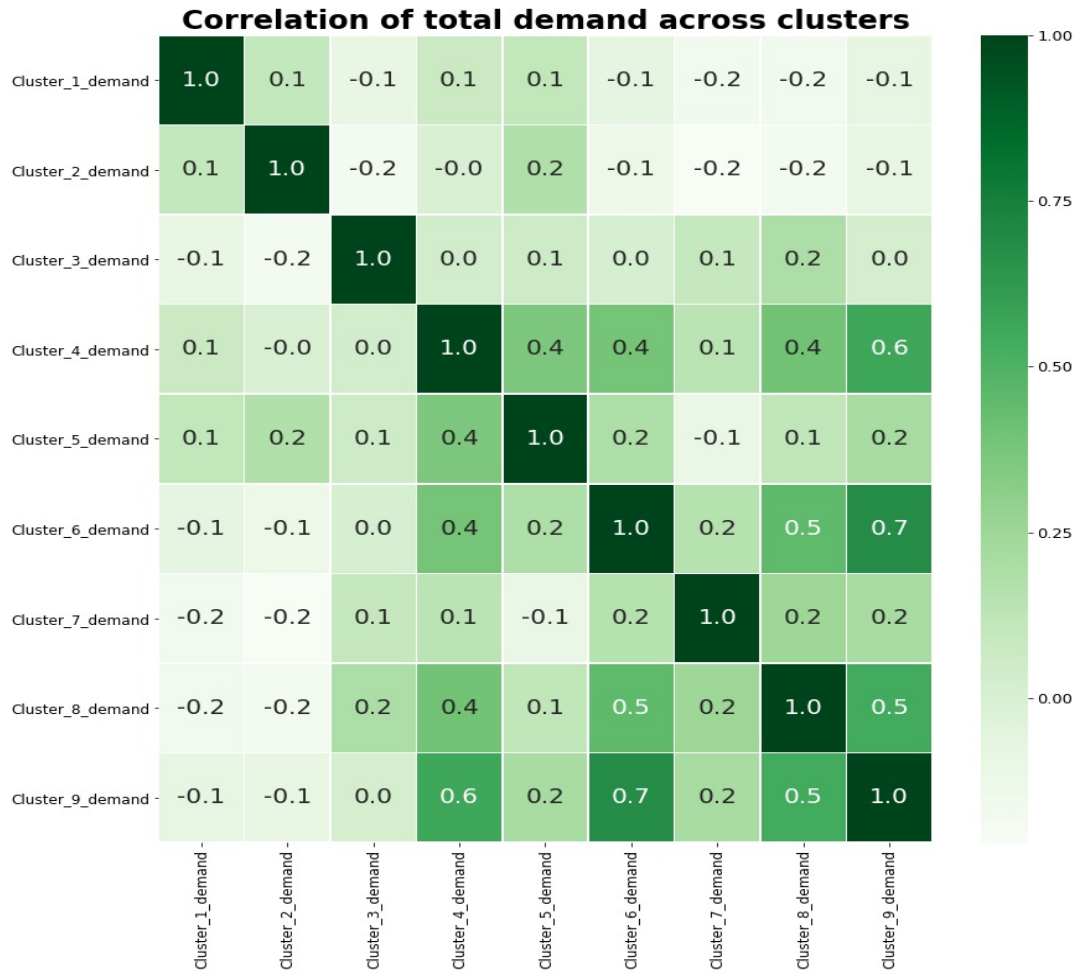

Figure 14 **Correlation heatmap for 9 clusters**

We can observe from the sample correlation heatmap above that there exist strong correlation between certain clusters (e.g cluster 6 and 9). To exploit the spatial correlation effects, these are the steps taken to predict the cluster total demands:

1. We first treat each cluster as a variable, where each cluster has a time series which is a vector of demands, calculated as the sum of the demand of all locations in the cluster, as shown above as $D_{C(i)}$.

2. Next, we check for stationarity for all the cluster's time series and perform differencing if they are non-stationary.

3. We then fit a VAR model that uses 6 past time lags. Alternatively, we also fitted a STAR model that uses 6 past time lags too.

4. If we have performed differencing in step 2, we then use the VAR/STAR model to predict the differenced values for time-step 199 to 204. We then add back the difference to the previous values to convert it back to the original demand for each cluster.

If we want to use a VARX model,the exogenous variables of each locations would have to be aggregated in some way when we combine the different locations into clusters, which might introduce a additional source of error. Hence, in this section, we would only use VAR/STAR since it does not require any form of aggregation or manipulation of exogenous variables.

9.3. Assigning total cluster demand to individual location

After we attained the total demand for each cluster, the next step would be to reallocate the total demand to each individual location in the cluster, making sure that the distribution of demand across the locations are relatively similar to as before.

One assumption that was made here is that the distribution of the demand across the locations in each cluster remain relatively constant over time. Taking the previous distribution into account, the predicted demand for each cluster would then be reallocated to each individual location in the cluster using this equation:

$$y_i = Y_{C(i)} * \frac{(\sum_{j=1}^{n} x_{ij})}{D_{C(i)}} \forall\, i$$

where y_i represents the predicted demand for location i, $Y_{C(i)}$ represents the predicted total demand of the respective cluster of location i, $C(i)$ represents the cluster which location i belongs to, $D_{C(i)}$

represents the total demand (from training set) of the cluster C(i) and x_{ij} represents the past number orders at location i at time j.

9.4. Results

In this 3-step approach, it is not appropriate to use conventional methods, such as the elbow method, to determine the number of clusters. This is because our purpose here is not to minimize the within cluster sum-of-squares, but rather to find the optimal combination of groups that yields the best forecast accuracy after we fit a VAR or a STAR model on them and reallocate the demand using the VAR/STAR model predictions. Hence, we tried different number of clusters for the 3 different distance metric as mentioned in Section 9.1 and observe the MSE and quadratic score for each of them.

9.5. MSE result from using VAR to make cluster predictions for different cluster size and distance metrics

Cluster Size	Euclidean Distance	Correlation between time series	DTW Distance
2	69.71	62.83	62.96
3	60.11	61.97	61.49
4	62.09	65.51	61.28
5	60.28	64.31	61.00
6	63.35	63.95	61.40
7	61.68	65.12	60.73
8	**58.19**	63.13	**60.36**
9	63.27	64.50	60.55
10	62.55	**58.48**	61.37
11	59.53	64.26	63.44
12	61.38	65.39	66.15

Table 1 MSE result from using VAR to make cluster predictions for different cluster size and distance metrics

9.6. Quadratic Score from using VAR to make cluster predictions for different cluster size and distance metric

Cluster Size	Euclidean Distance	Correlation between time series	DTW Distance
2	-785.07	-786.50	-785.32
3	-786.93	**-786.66**	-785.45
4	-786.41	-786.06	-785.70
5	-786.44	-761.14	-785.75
6	-776.18	-741.99	-785.67
7	-776.14	-724.84	**-785.88**
8	**-787.15**	-694.31	-785.25
9	-741.26	-693.53	-785.45
10	-777.10	-678.20	-784.90
11	-776.79	-653.50	-784.61
12	-785.73	-737.14	-784.37

Table 2 Quadratic Score from using VAR to make cluster predictions for different cluster size and distance metric

9.7. MSE result from using STAR to make cluster predictions for different cluster size and distance metrics

Cluster Size	Euclidean Distance	Correlation between time series	DTW Distance
2	61.92	68.10	61.43
3	60.91	64.58	60.52
4	62.40	64.01	60.99
5	61.14	**61.35**	60.34
6	60.83	64.03	61.38
7	**58.95**	63.86	60.83
8	63.03	66.87	59.42
9	62.06	67.95	59.00
10	59.08	61.87	58.73
11	61.25	63.85	**58.53**
12	60.15	64.60	59.37

Table 3 MSE result from using STAR to make cluster predictions for different cluster size and distance metrics

9.8. Quadratic Score from using STAR to make cluster predictions for different cluster size and distance metric

Cluster Size	Euclidean Distance	Correlation between time series	DTW Distance
2	-786.57	-786.48	-786.91
3	-753.94	-786.15	-786.97
4	-758.61	-786.65	-786.97
5	-757.77	**-787.25**	-787.09
6	**-787.01**	-740.66	-786.93
7	-760.05	-755.22	-787.00
8	-760.44	-699.55	-787.09
9	-752.48	-746.15	-787.23
10	-732.82	-728.37	**-787.25**
11	-738.39	-682.46	-787.19
12	-741.95	-647.06	-787.21

Table 4 Quadratic Score from using STAR to make cluster predictions for different cluster size and distance metric

9.9. Plots of results

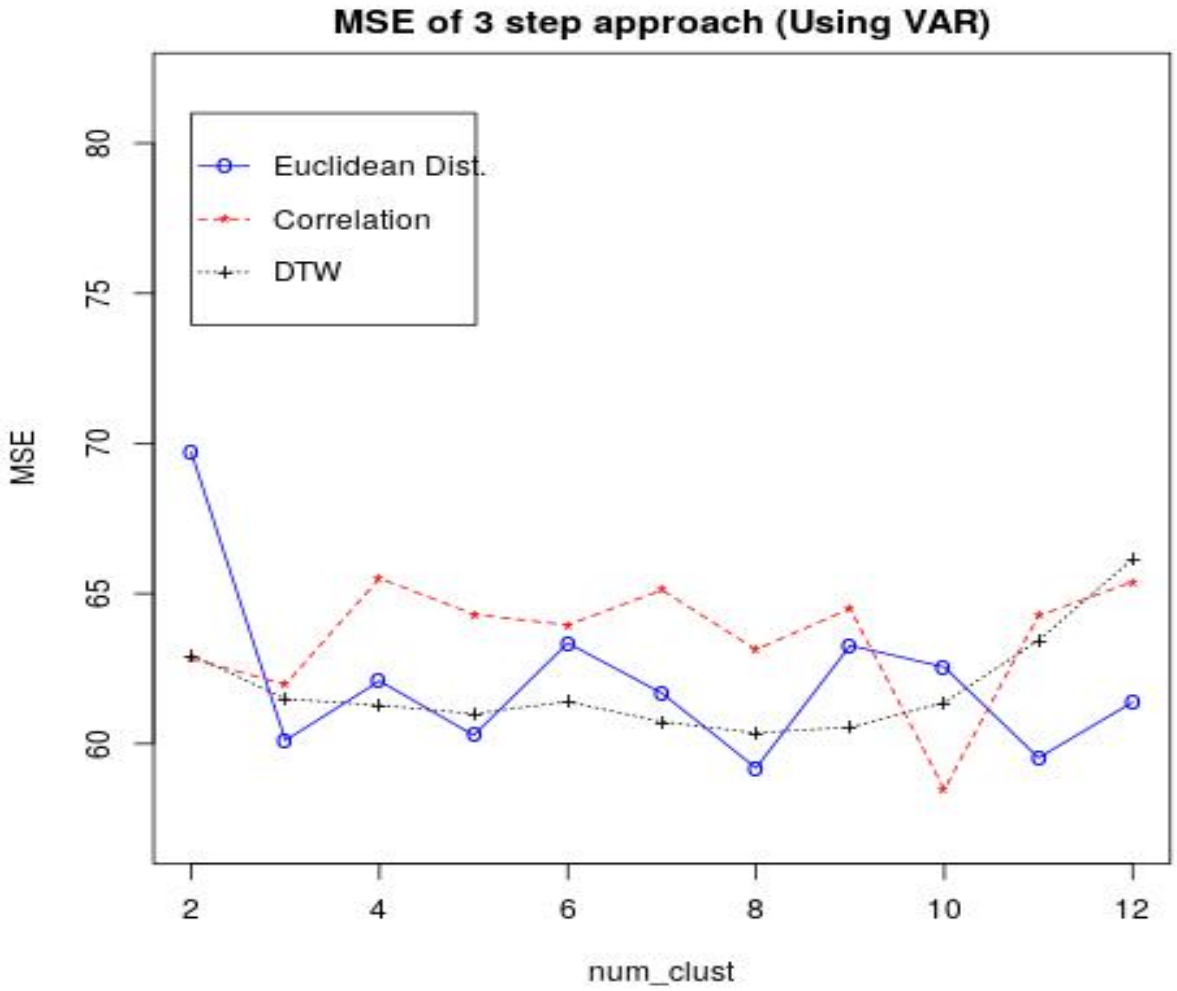

Figure 15 **Plot of MSE using the 3-step approach which uses VAR to predict cluster demands. We can see that the optimal choice based on MSE would be using the euclidean distance of the locations with 8 clusters, giving a MSE of 58.19**

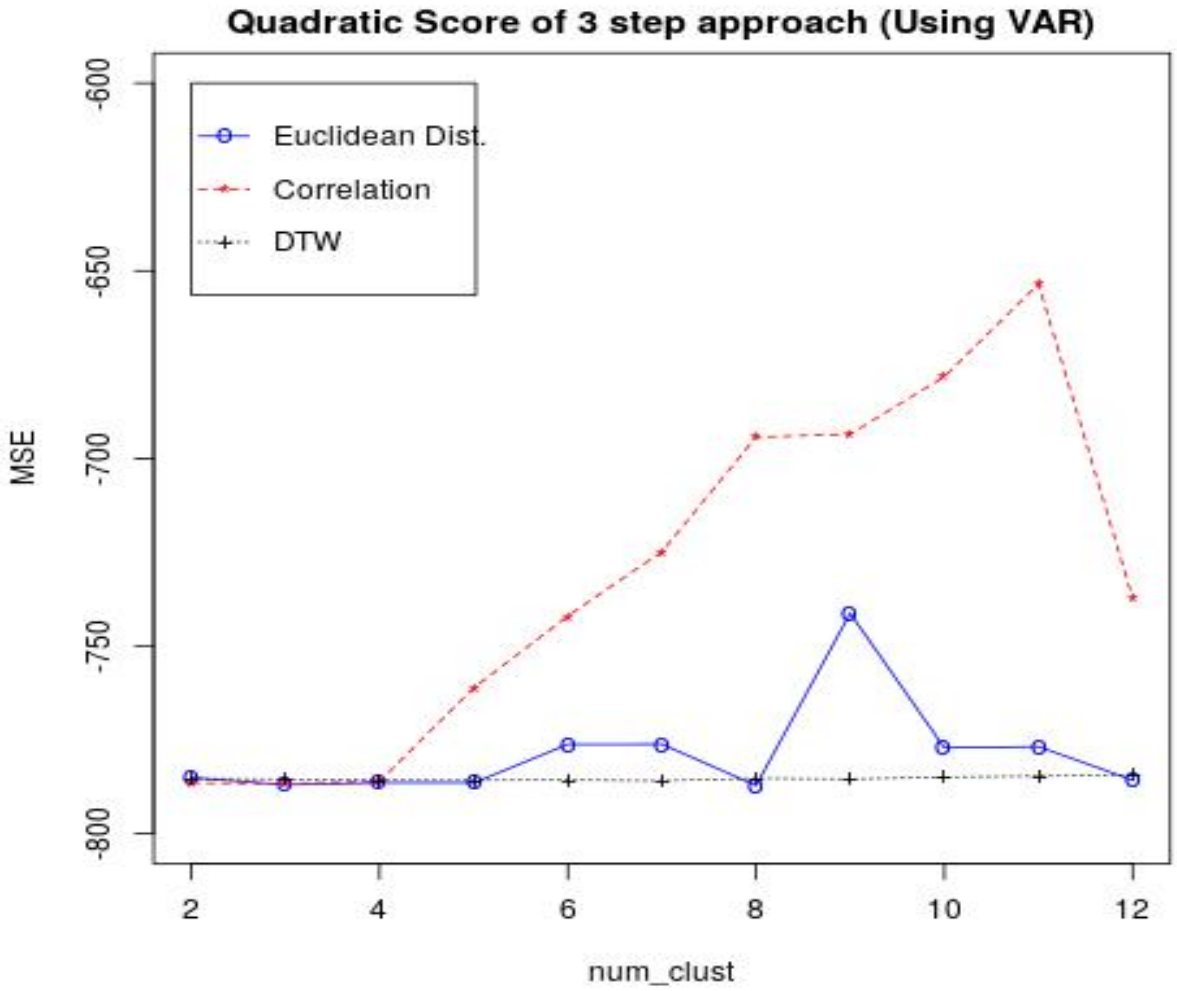

Figure 16 **Plot of Quadratic score using the 3-step approach which uses VAR to predict cluster demands. We can see that the optimal choice based on quadratic score is using euclidean distance of the locations with 8 custers, giving a quadratic score of -787.15**

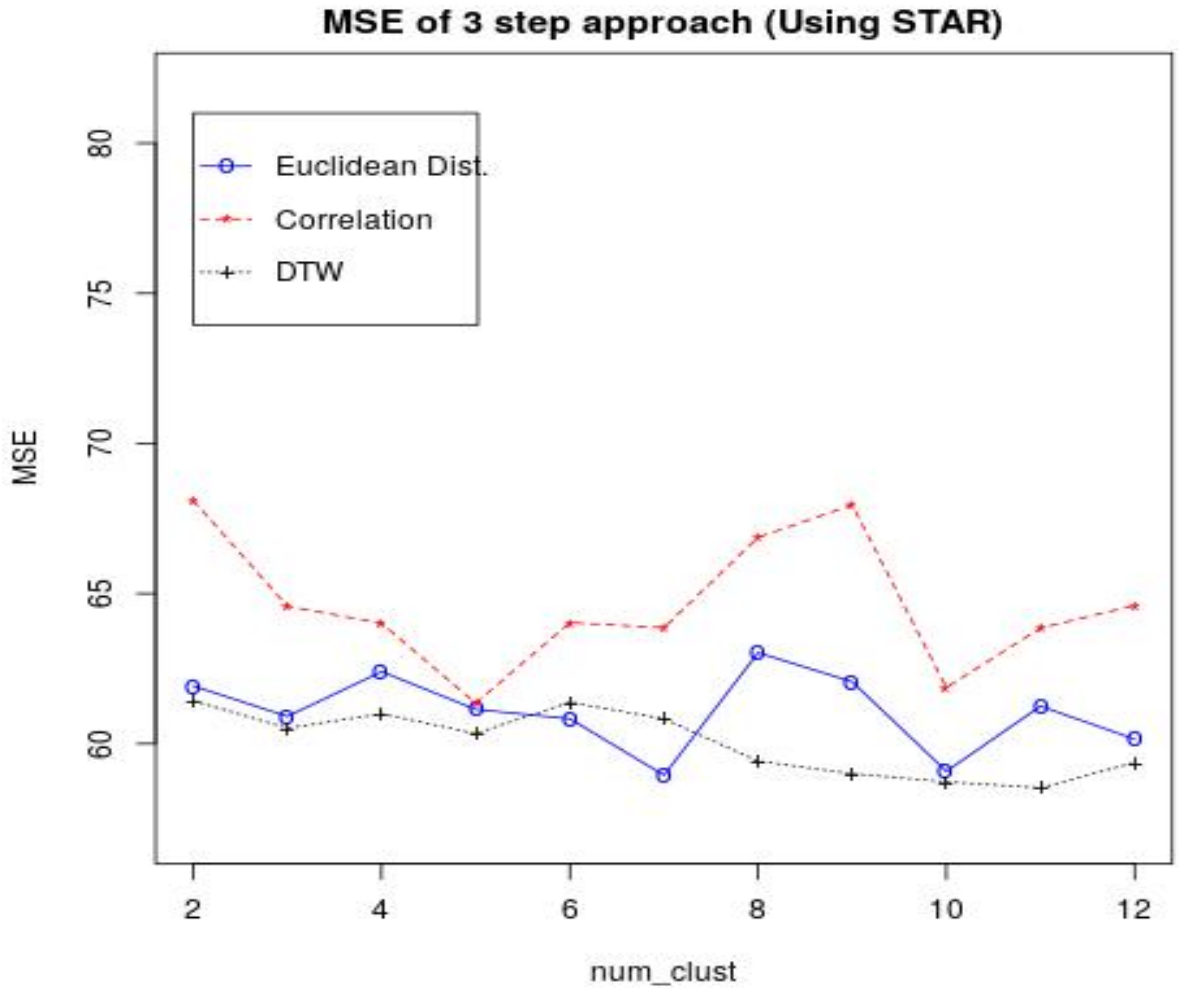

Figure 17 **Plot of MSE using the 3-step approach which uses STAR to predict cluster demands. We can see that the optimal choice based on MSE would be using the DTW between time series with 11 clusters, giving a MSE of 58.53**

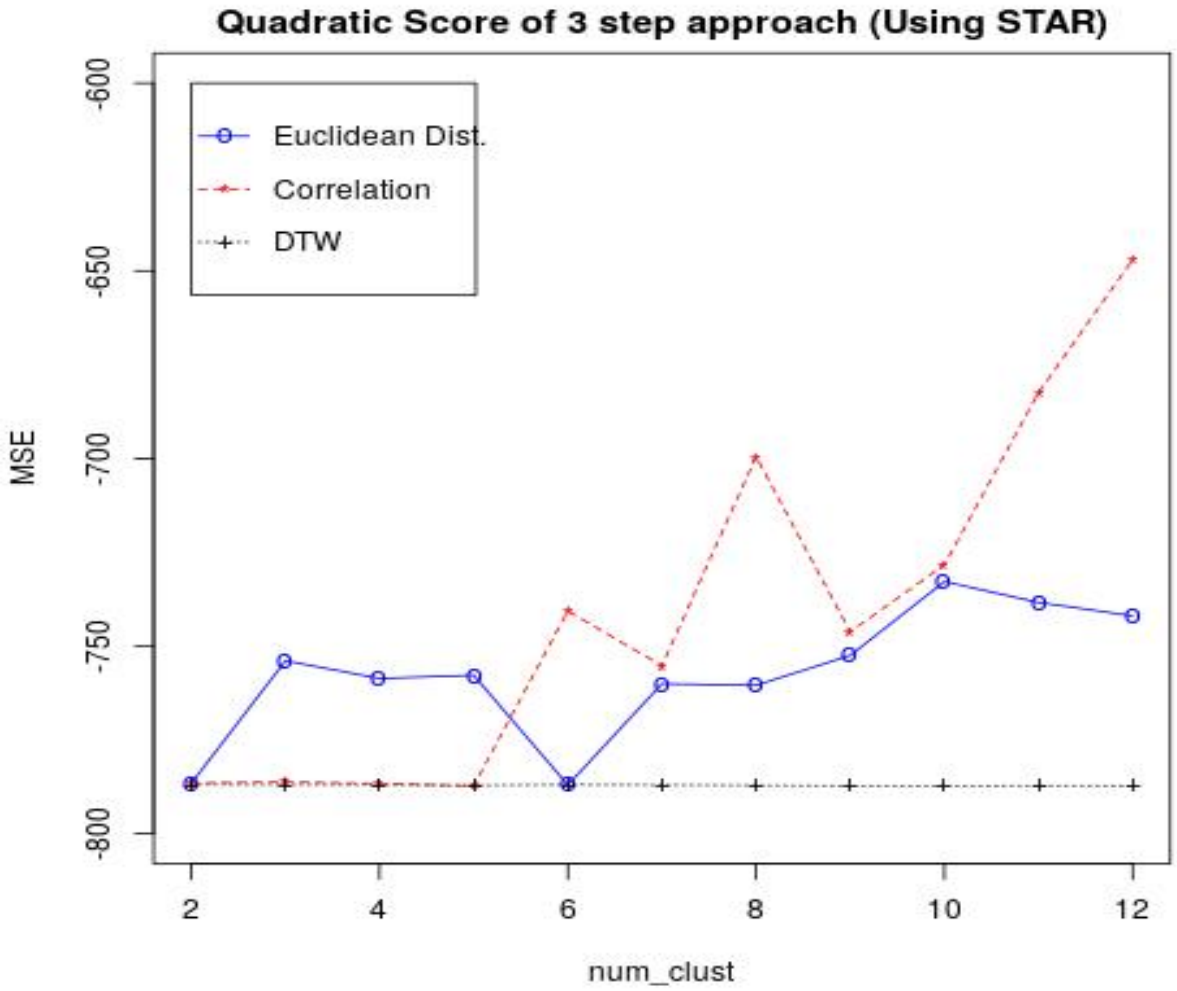

Figure 18 **Plot of Quadratic score using the 3-step approach which uses STAR to predict cluster demands. We can see that the optimal choice based on quadratic score is using DTW between time series with 10 custers, giving a quadratic score of -787.25**

Based on the results, if we prioritised MSE as the main metric, the optimal result could be obtained by clustering the time series into 8 groups using the euclidean distance between their physical locations as the distance metric, use VAR to predict cluster demands, and reassign it back to individual locations to give the lowest MSE of **58.19**.

10. Results and Conclusion

The table below summarises the results of our models.

	Small Dataset		Full Dataset	
	MSE	**Quad Score**	**MSE**	**Quad Score**
ARIMA	46.87	-19.12	71.73	-772.45
GLM	38.16	-19.90	82.42	-656.94
VAR	47.28	-15.11	82.87	-750.94
VARX	45.76	-16.23	73.69	-727.02
STAR	39.76	-20.27	81.50	-656.32
3-step approach (geographical location,VAR)	39.46	-19.28	**58.19**	-787.15
3-step approach (geographical location,STAR)	37.26	-16.98	58.95	-760.05
3-step approach (correlation of time series,VAR)	40.73	-18.28	58.48	-678.20
3-step approach (correlation of time series,STAR)	36.81	-19.53	61.35	-787.25
3-step approach (DTW of time series,VAR)	37.96	-19.69	60.36	-785.25
3-step approach (DTW of time series,STAR)	35.16	-19.70	58.53	-787.19

Table 5 MSE and Quadratic Score of the different methods used in this paper. The MSE is prioritised here, and hence the quadratic score might not be the optimal one.

While both MSE and Quadratic scores are important metrics to evaluate our models, it is often hard to find a model that has the minimum for both metrics. In this paper, we would prioritise the MSE due to its predictive performane measure. The optimal model would be to use the 3-step approach with the euclidean distance between geographical locations as the distance metric, and using VAR to predict the clusters' total demand. We can also observe that while the Quadratic score for this model is not the minimum, it is the second minimum, with only a very minute difference of 0.1 (-787.15 vs -787.25).

Our STAR and VARX models performs better than the GLM models on the full dataset, suggesting that the spatial relationship between locations are useful in forecasting. Also, VARX performs better than VAR model in both datasets, suggesting the exogenous variables have some explanatory power and do improve the forecast accuracy. Applying the 3-step approach also gives a much improved result as compared to just using VAR/VARX or STAR models on all the locations. This might be due to the 3-step approach being more reasonable in dealing with locations with sparse demand due to the aggregation of clusters in the clustering method. In the 3-step approach, our VAR model uses just the 8 clusters as variables to predict the total cluster demand for just 8 clusters, while the original VAR uses all 839 locations as variables to predict for

839 locations, which is more likely to produce a higher error rate.

As for the distance metric used for clustering, using euclidean distance between the geographical location gives us the best result. However, using correlation or DTW between the time series to cluster also gives similar results and that would imply that spatial correlation might not only be limited to just purely geographical proximity but also on the time series similarity. Comparing correlation measure and DTW measure, using correlation between the time series as the distance metric gives us the better result. Correlation compares the general shape and trend of the time series regardless of scale, whereas for DTW, it is more effective to compare time series of varying length and speed.

11. Conclusion and Future Work

This paper have attempted to tackle the 3 main challenges of this dataset as mentioned in section 3.3. We have introduced a 3-step approach model to deal with sparse data and counts across multiple locations. The clustering step in the 3-step approach have also attempted to reduce the dimensionality (number of locations used as variables) of our dataset. We have also used probabilistic forecasting methods as well as proper scoring metrics to evaluate our models for count data. For practical purposes, our fractional numbers that is obtained from our predictions could be rounded up to the nearest integer since the nearest integer would be the count value that is expected to occur.

Our 3-step approach model display decent MSE result. However, using VAR is still technically a linear model and future work could include exploring using non-linear models like neural network, such as Long Short-Term Memory (LSTM) models or Convolutional LSTM models that also uses spatial-temporal features. Neural network models are also believed to be able to take into account the non-stationarity of the time series, which would allow our model to be more flexible without the need for differencing.

Another area that could be explored further would be the method used to assign the total cluster demand to individual locations. Our current method simply used the historical distribution of the total sum of demand of each location over all the locations in that cluster. This would work well if the distribution of the demand for the locations remain relatively constant over time but if the distribution of the locations fluctuates greatly, the current reassignment formula might not be a suitable model. Instead, we could attempt to create another model to predict the distribution of the locations in each cluster in order to get a relatively more stable and reasonable distribution over time.

12. Appendix

12.1. Distribution Plots of Exogenous Variables

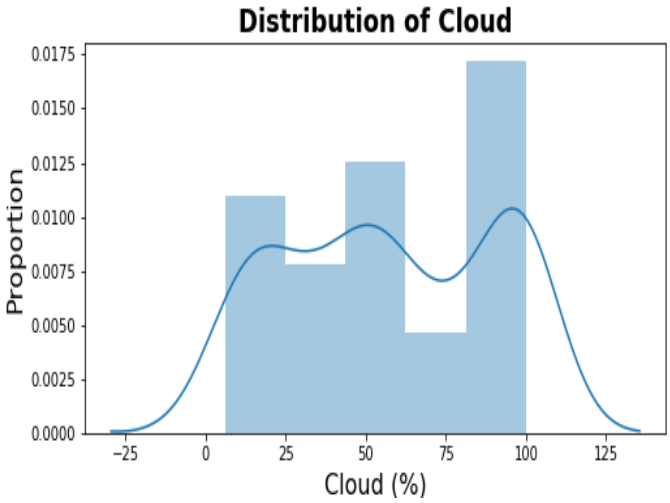

Figure 19 **Distribution of Cloud**

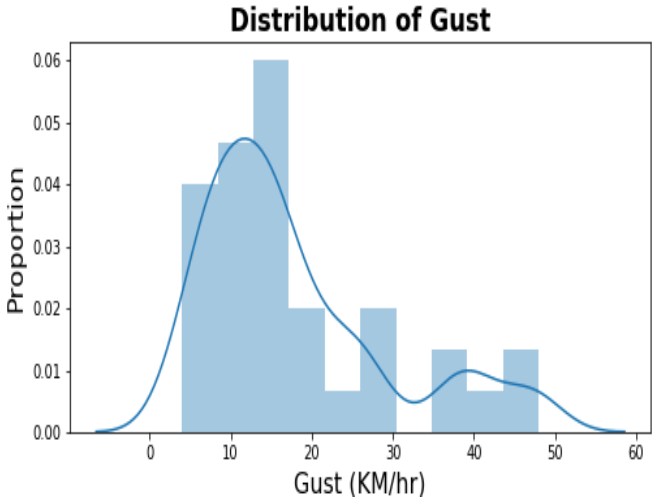

Figure 20 **Distribution of Gust**

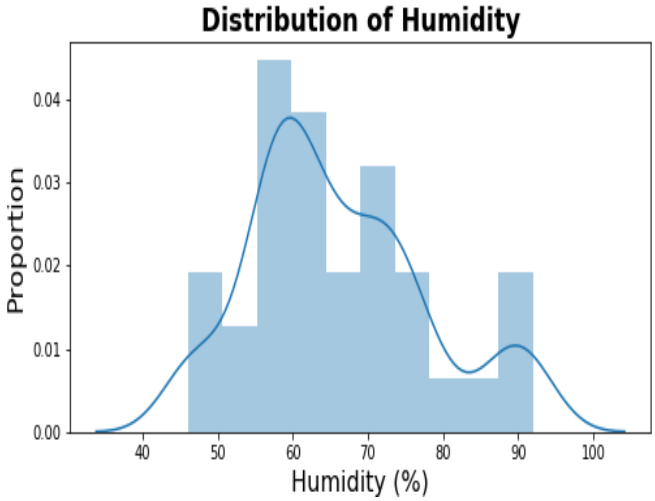

Figure 21 **Distribution of Humidity**

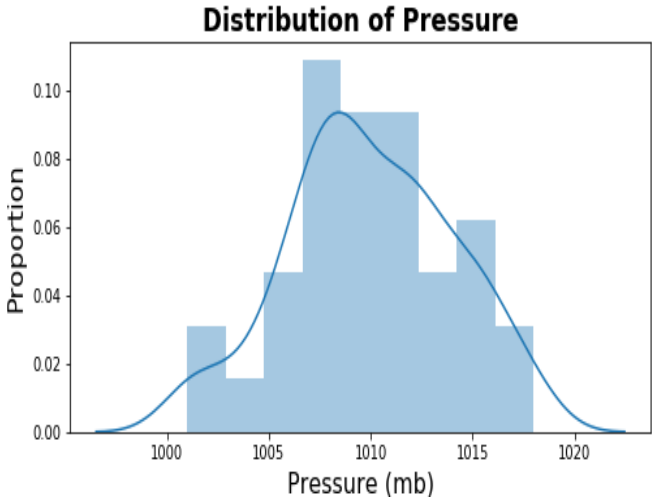

Figure 22 **Distribution of Pressure**

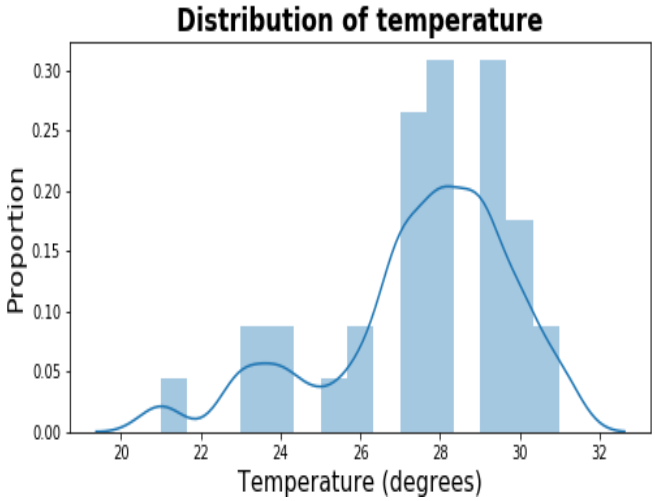

Figure 23 Distribution of Temperature

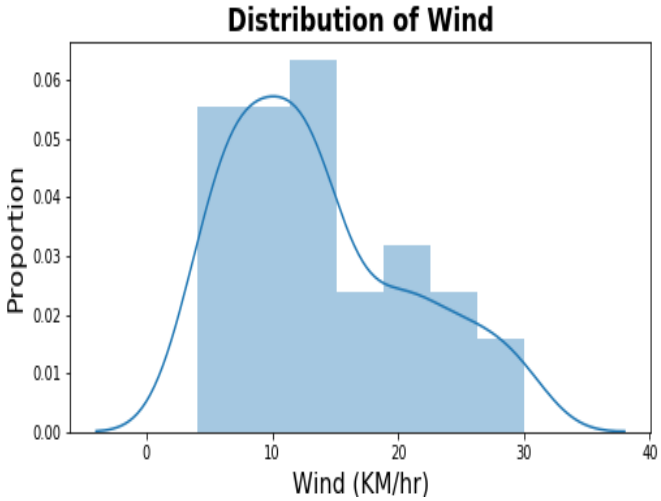

Figure 24 Distribution of Wind

12.2. Mean Demand Against Exogenous Variables

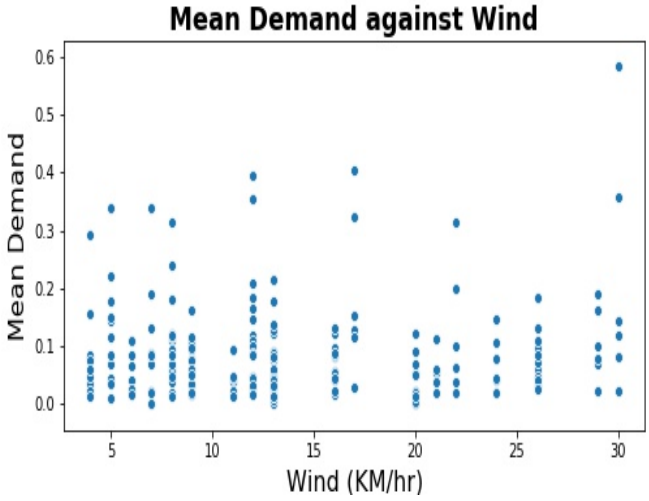

Figure 25 **Mean Demand against Wind**

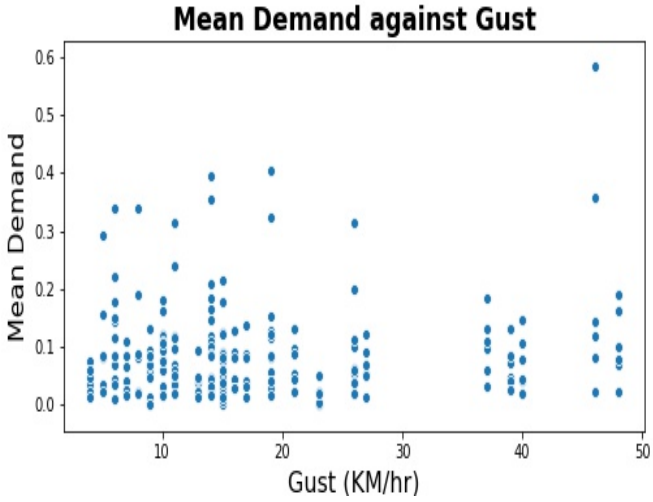

Figure 26 **Mean Demand against Gust**

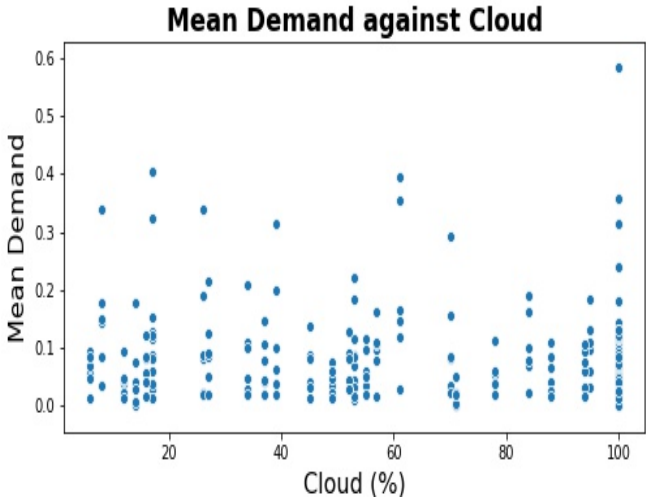

Figure 27 Mean Demand against Cloud

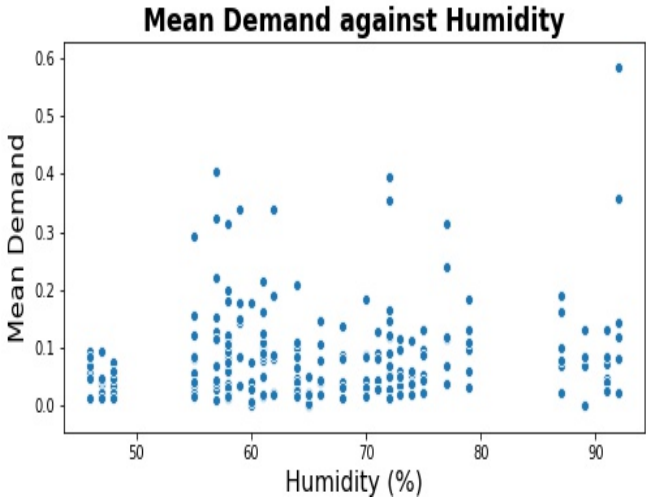

Figure 28 Mean Demand against Humidity

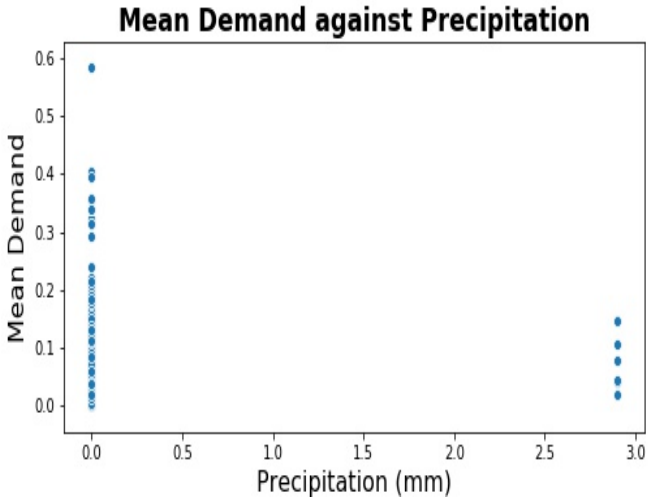

Figure 29 Mean Demand against Precipitation

References

Baumöhl, Eduard, Štefan Lyócsa. 2009. Stationarity of time series and the problem of spurious regression. *SSRN Electronic Journal* doi:10.2139/ssrn.1480682.

Bjørnland, Hilde Christiane. 2014. Var models in macroeconomic research.

Box, George E. P., George C. Tiao. 1977. A canonical analysis of multiple time series.

Czado, Claudia, Tilmann Gneiting, Leonhard Held. 2009. Predictive model assessment for count data. *Biometrics* **65 4** 1254–61.

Ding, Hui, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, Eamonn Keogh. 2008. Querying and mining of time series data: Experimental comparison of representations and distance measures. *Proc. VLDB Endow.* **1**(2) 1542–1552. doi:10.14778/1454159.1454226. URL `https://doi.org/10.14778/1454159.1454226`.

Engle, Robert F, Clive W J Granger. 1987. Co-integration and Error Correction: Representation, Estimation, and Testing. *Econometrica* **55**(2) 251–276. URL `https://ideas.repec.org/a/ecm/emetrp/v55y1987i2p251-76.html`.

Farhath, Z. Asha, B. Arputhamary, Dr. L. Arockiam. 2016. A survey on arima forecasting using time series model.

Fattah, Jamal, Latifa Ezzine, Zineb Aman, Haj Moussami, Abdeslam Lachhab. 2018. Forecasting of demand using arima model. *International Journal of Engineering Business Management* **10** 184797901880867. doi:10.1177/1847979018808673.

Johansen, Soren. 1991. Estimation and Hypothesis Testing of Cointegration Vectors in Gaussian Vector Autoregressive Models. *Econometrica* **59**(6) 1551–1580. URL `https://ideas.repec.org/a/ecm/emetrp/v59y1991i6p1551-80.html`.

Knight, Marina I., M. A. Nunes, Guy P. Nason. 2016. Modelling, detrending and decorrelation of network time series.

Kurt, Serkan, K. Batu Tunay. 2015. Starma models estimation with kalman filter: The case of regional bank deposits. *Procedia - Social and Behavioral Sciences* **195** 2537 – 2547. doi:https://doi.org/10.1016/j.sbspro.2015.06.441. URL `http://www.sciencedirect.com/science/article/pii/S1877042815039208`. World Conference on Technology, Innovation and Entrepreneurship.

Liboschik, Tobias, Konstantinos Fokianos, Roland Fried. 2017. tscount: An r package for analysis of count time series following generalized linear models. *Journal of Statistical Software, Articles* **82**(5) 1–51. doi:10.18637/jss.v082.i05. URL `https://www.jstatsoft.org/v082/i05`.

Liu, Sheng, Long He, Zuo-Jun Max Shen. 2020. On-time last mile delivery: Order assignment with travel time predictors. *Management Science* .

Lu, Chi-Jie, Chi-Chang Chang. 2014. A hybrid sales forecasting scheme by combining independent component analysis with k-means clustering and support vector regression. *TheScientificWorldJournal* **2014** 624017. doi:10.1155/2014/624017.

Luna, Xavier, Marc Genton. 2005. Predictive spatio-temporal models for spatially sparse environmental data. *Statistica Sinica* **15** 547–568.

Molenaar, Dylan, Maria Bolsinova. 2017. A heteroscedastic generalized linear model with a non-normal speed factor for responses and response times. *British Journal of Mathematical and Statistical Psychology* **70**. doi:10.1111/bmsp.12087.

Murray, Paul, Bruno Agard, Marco Barajas. 2015. Forecasting supply chain demand by clustering customers. *IFAC-PapersOnLine* **48** 1834–1839. doi:10.1016/j.ifacol.2015.06.353.

Mushtaq, Rizwan. 2011. Augmented dickey fuller test. *SSRN Electronic Journal* doi:10.2139/ssrn.1911068.

Nelder, J. A., R. W. M. Wedderburn. 1972. Generalized linear models. *Journal of the Royal Statistical Society. Series A (General)* **135**(3) 370–384. URL `http://www.jstor.org/stable/2344614`.

Nicholson, William, David Matteson, Jacob Bien. 2017. Bigvar: Tools for modeling sparse high-dimensional multivariate time series .

Nunes, M.A., M.I. Knight, G.P. Nason. 2015. *Modelling and prediction of time series arising on a graph*. Lecture Notes in Statistics - Proceedings, Springer International Publishing, 183–192. doi:10.1007/978-3-319-18732-7.

Safikhani, Abolfazl, Camille Kamga, Sandeep Mudigonda, Sabiheh Faghih, Bahman Moghimi. 2017. Spatio-temporal modeling of yellow taxi demands in new york city using generalized star models. *International Journal of Forecasting* doi:10.1016/j.ijforecast.2018.10.001.

Senin, Pavel. 2008. Dynamic time warping algorithm review.

Tsay, R.S. 2013. *Multivariate Time Series Analysis: With R and Financial Applications*. Wiley Series in Probability and Statistics, Wiley. URL `https://books.google.com.sg/books?id=A4QVAgAAQBAJ`.

Wang, Dongjie, Yan Yang, Shangming Ning. 2018. Deepstcl: A deep spatiotemporal convlstm for travel demand prediction. doi:10.1109/IJCNN.2018.8489530.

Wecker, William E. 1989. Comment: Assessing the accuracy of time series model forecasts of count observations. *Journal of Business & Economic Statistics* **7**(4) 418–419. URL `http://www.jstor.org/stable/1391640`.

Xiao, Zhijie, Peter C. B. Phillips. 1998. An adf coefficient test for a unit root in arma models of unknown order with empirical applications to the us economy.