NUS Business School Honors Dissertation

Peng Seng Ang

Advised by: Professor He Long

AY 2019/2020 Semester 2

Abstract

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 instead count variables, like number of orders and number of transactions. Previous research have mainly explored neural network methods for spatio-temporal problems. In this paper, we focus on how we can apply classical time series model, using Vector Autoregressive models to exploit spatial correlations as well as introduce a 3-step approach to improve forecast accuracy.

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. The motivation for this paper comes from [19], where the focus is to optimise assignments of lunch delivery orders to drivers in order to minimize the total delay of all drivers. The dataset used in both [19] 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.

In reality, demand is never deterministic and hence, having an accurate forecast of demand for the food service providers would help them more effectively and efficiently assign orders to drivers to improve the overall delivery time. As such, 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, as well as Vector Autoregressive (VAR) models, that include both spatial and temporal features would be explored. We also introduce a 3-step approach that combined 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 [7], where they applied deep spatio-temporal convolutional LSTM methods for traffic demand prediction.

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 [1], where they proposed using generalized spatio-temporal autoregressive model for predicting taxi demands across locations in New York City.

[12] 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 [13] since the neighbouring counties does not provide a substantial amount of explanatory power. Other related literature includes [6] where they propose a model building strategy for spatially sparse but temporally rich data.

BigVAR and tscount are some libraries that would be used in this paper. [20] provides the mathematical background and implementation of Generalised Linear Models (GLM) for count time series as a library (tscount) in R while [22] 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 https://www.worldweatheronline.com/shanghai-weather-history/shanghai/cn.aspx was used to include weather and rainfall data as well as encoding of the weekadys for all the respective days.

3.1 Exploratory Analysis

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

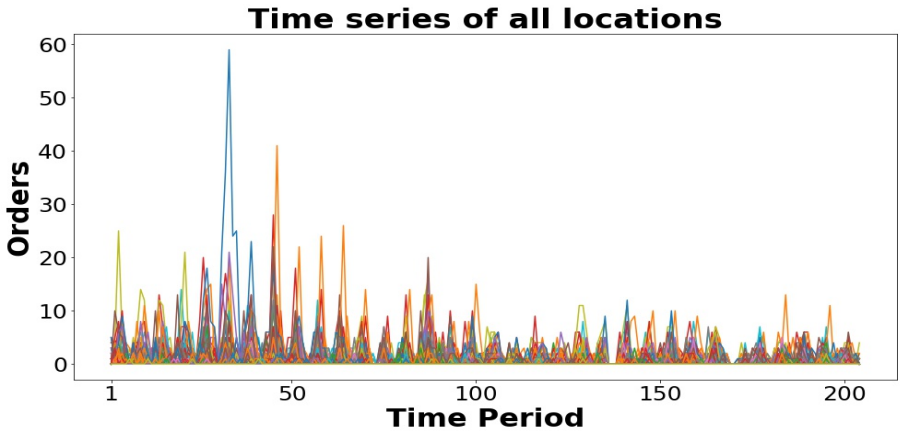

Figure 1: Time series of all locations in the dataset. It can be observed that most locations have very low number of orders across the time period.

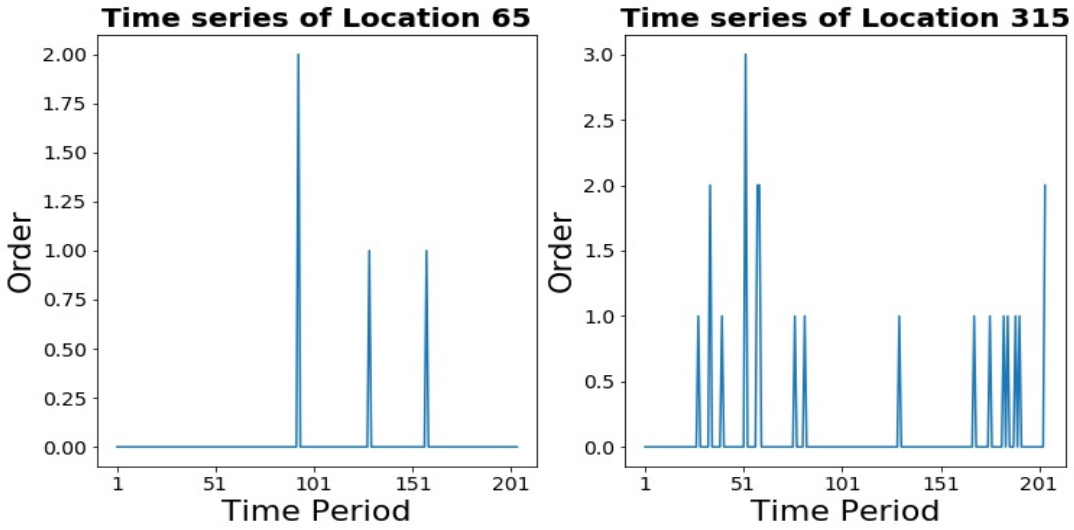

Figure 2: Most locations have very sparse time series (left) while some have relatively more dense time series (right)

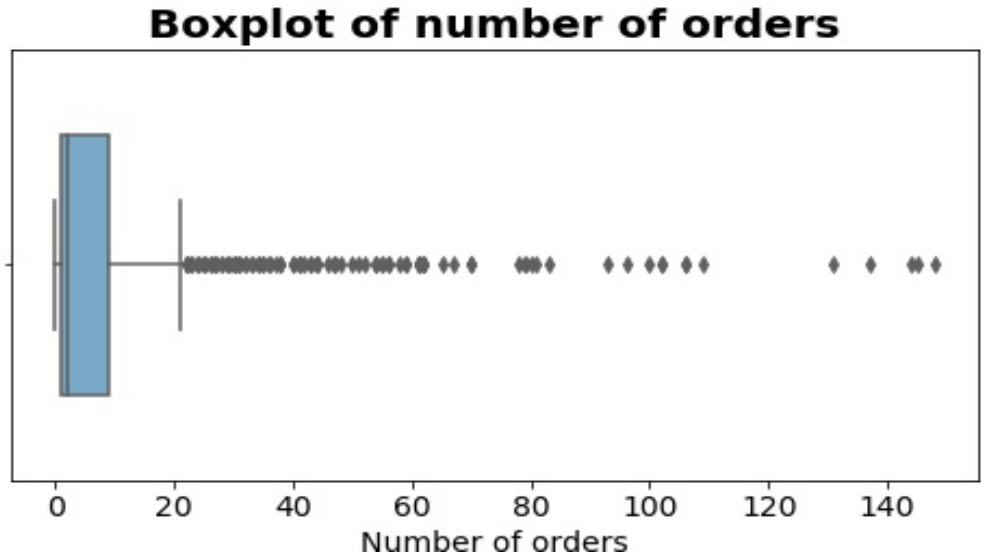

Figure 3: Boxplot of counts. We can see that most of the locations have extremely low number of non-zero orders and further analysis showed that about 335 locations have just a maximum of one non-zero order throughout the 204 time periods.

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

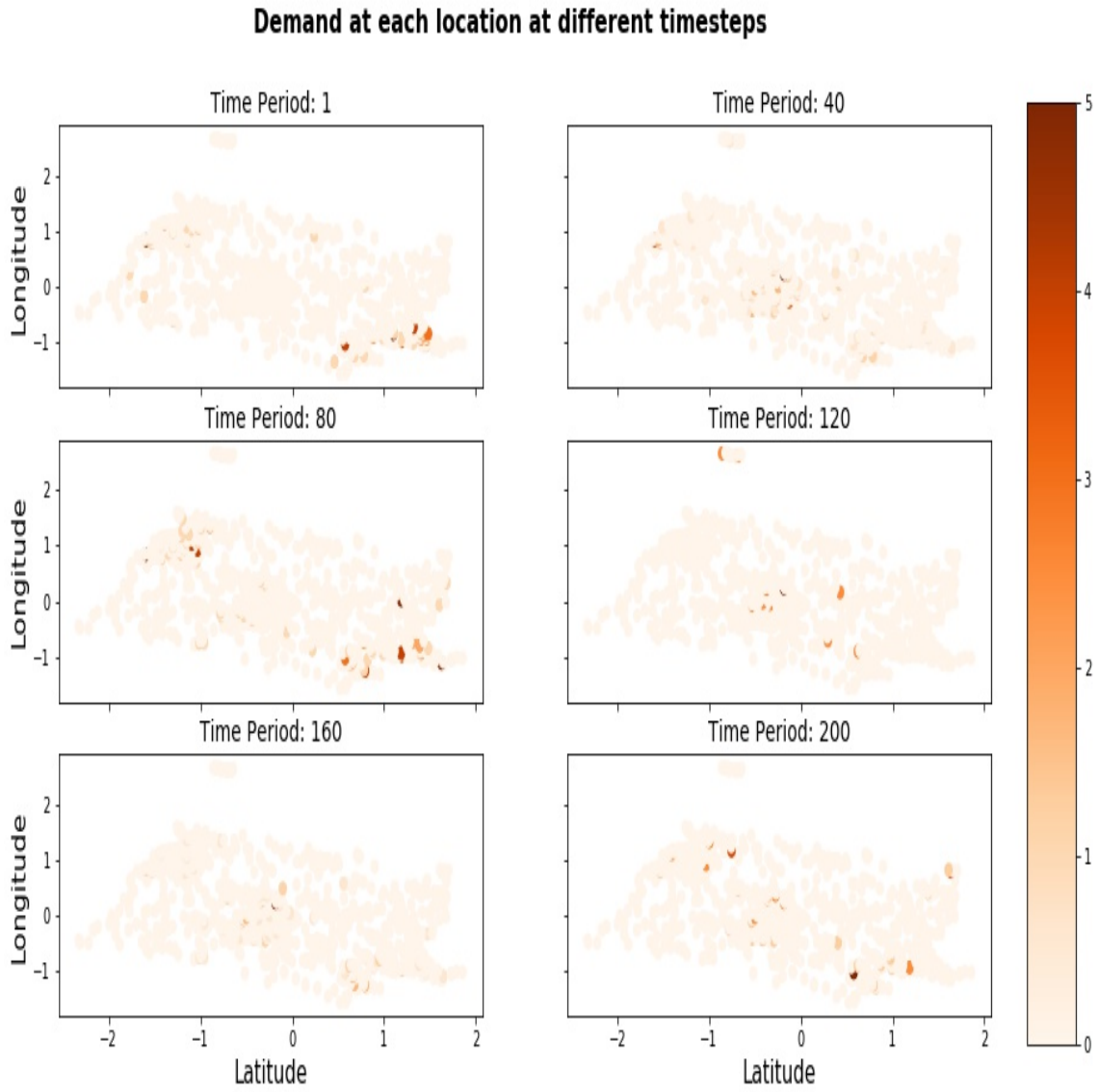

Figure 4: Demand across locations over time. Plot suggests that the areas towards the right tend to have higher demand.

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.

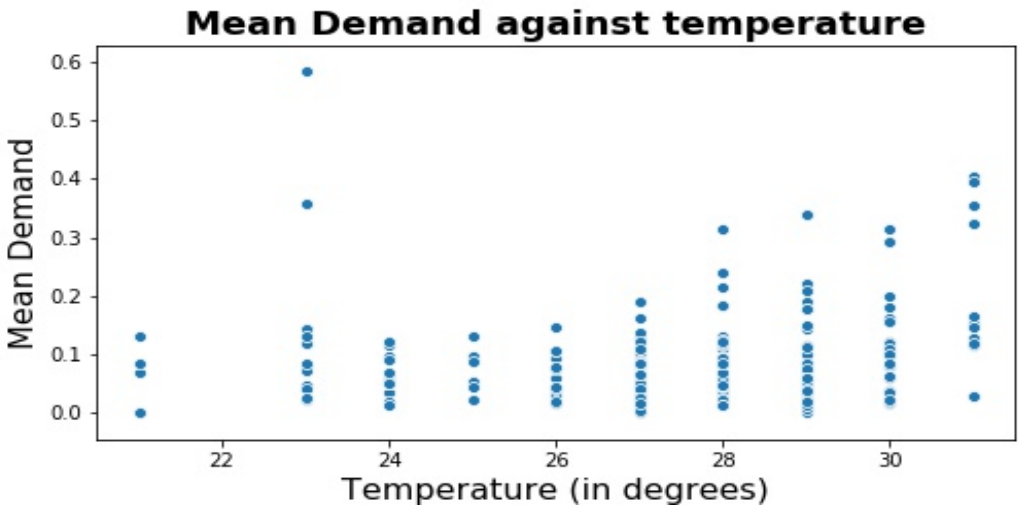

Figure 5: Scatter plot of mean demand against temperature.

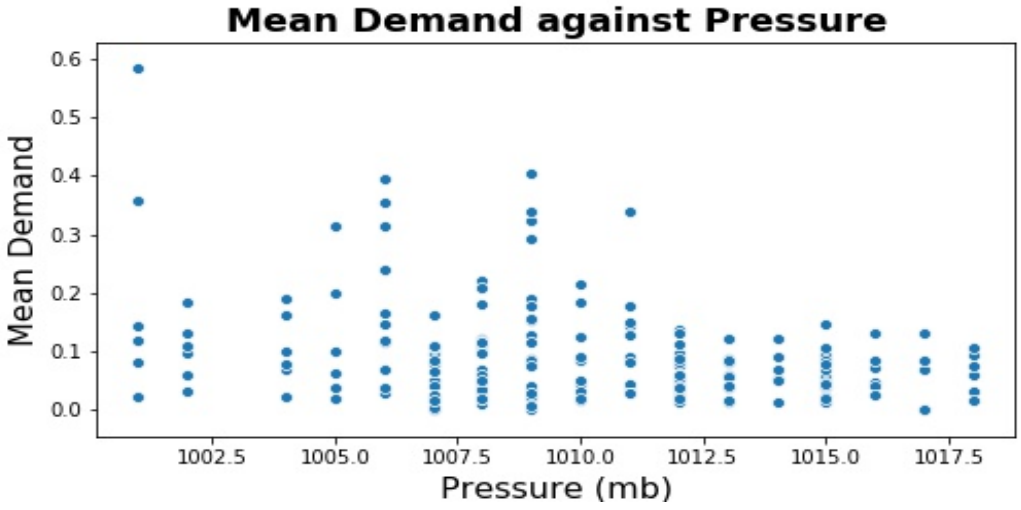

Figure 6: Scatter plot of mean demand against pressure

The scatter plot in Figure 4 visually display a slight positive relationship between tempera-

ture 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 full dataset to test and fit our model.

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 the day, hence only required to predict one-day ahead demand.

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

The following sections of the paper is structured as follows. We first implement a simple baseline ARIMA model in section 4. Section 5 and 6 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 5 instead and in section 6, we fit a Vector Autoregressive Model using all the locations as variables. In section 7, we introduce a 3 step modelling approach which would be more interpretable and would also perform the best out of all the models.

4 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 different spatial temporal time series models and compare the results against the baseline model.

4.1 Metric Used

The main metric that would be used for comparison would be Mean Squared Forecast Error (MSFE), which is calculated by:

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

where n is the number of data points, $\hat{y_t}$ is the predicted demand at time t and y_t is the actual demand at time t.

4.2 ARIMA models

Autoregressive Integrated Moving Average (ARIMA) models are one of the most commonly used models for time series ([23]). ARIMA models are made up of 3 processes, mainly the Autoregressive (AR) process, the Integrated (I) process and the Moving Average (MA) process ([10]). The AR process assumes that each observation can be expressed as a linear combination of its past values. An AR(x) process would mean using x 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. The Integrated Process states that the time series can undergo differencing to ensure that the series is stationary. A MA(x) process would mean using x number of past error observations. 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 differences 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.

4.3 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 Python was used to implement this. The out-of-sample MSFE for this baseline model on the smaller dataset is **47.60** and MSFE for dataset of all locations is **72.48**. A sample forecast plot is shown below:

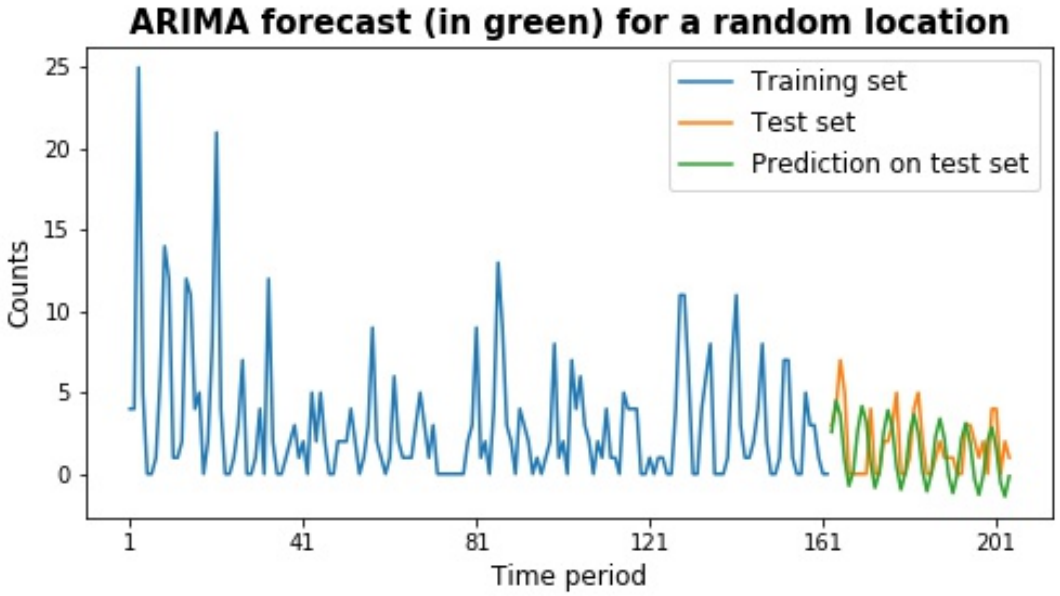

Figure 7: ARIMA forecast on a random location

5 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) [16]. GLM assumes that each value is conditionally dependent on its past values. According to [20], 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. X_t represents the a vector of the values of the previous time steps in the time series.

5.1 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 [20], 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. The out-of-sample MSFE for this baseline model on the smaller dataset is **38.16**, which is better than just using ARIMA on the same dataset.

However, when it is implemented on the full dataset, the MSFE of the GLM increases to **82.42**, which is worse than just applying ARIMA on every location, which gives a MSFE of 72.48.

5.2 Model Diagnostics

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

5.2.1 Residuals plots

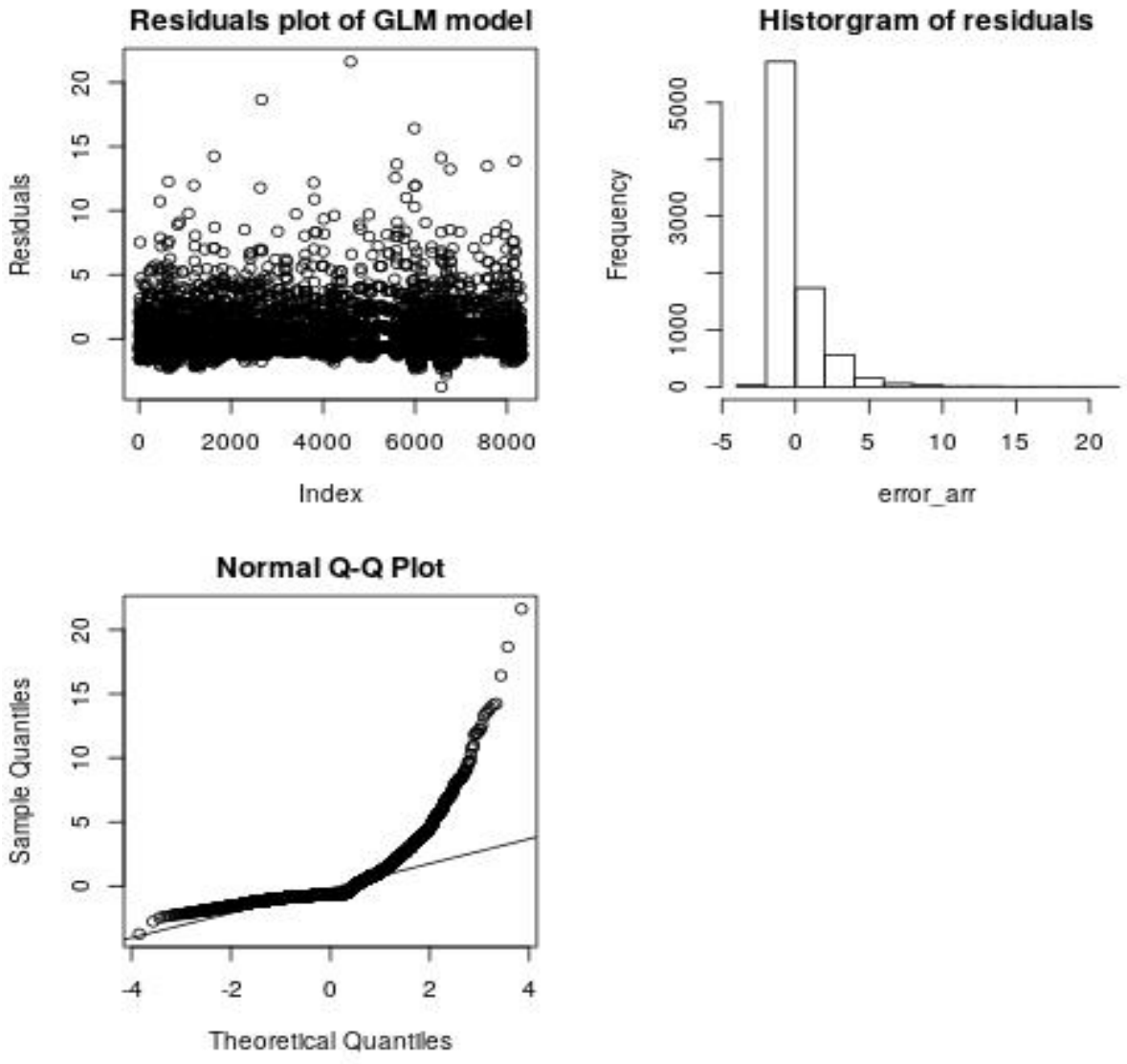

Figure 8: Residuals for GLM Model

The diagnostic plots in Figure 8 shows that while most residuals are randomly scattered around -5 to 5, there are still many points above 5, 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.

5.2.2 Residuals against Predicted values

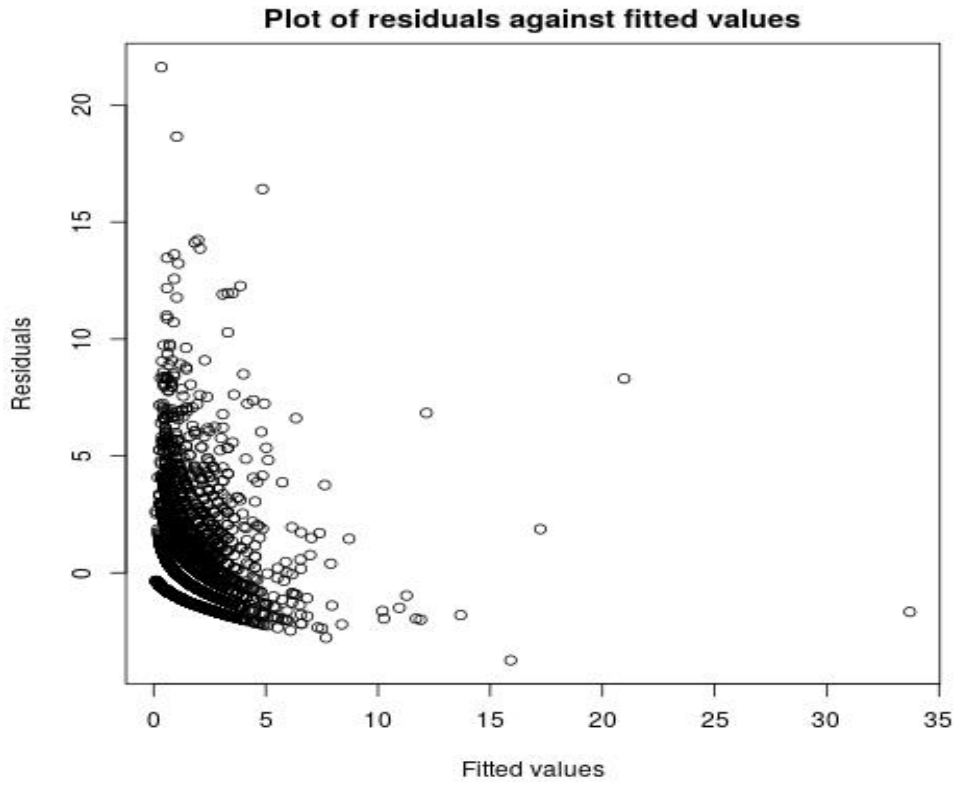

Figure 9: Residuals against Predicted values for GLM Model

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 [14].

5.3 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 next section explores another type of model which would use data from other locations as predictors.

6 Vector Autoregressive (VAR) Model

6.1 Details of VAR Model

Vector Autoregressive (VAR) models are the most commonly used model for multivariate time series, particularly in economics and financial time series as shown in [3]. VAR models are very similar to multivariate linear regression models and 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 if the time series can be written as:

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

where 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.

6.2 Motivation

As with many spatio-temporal demand prediction, spatial features could be an important feature if there exist spatial correlation. To check if there exist spatial correlation between the

locations, a correlation heatmap would be 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.

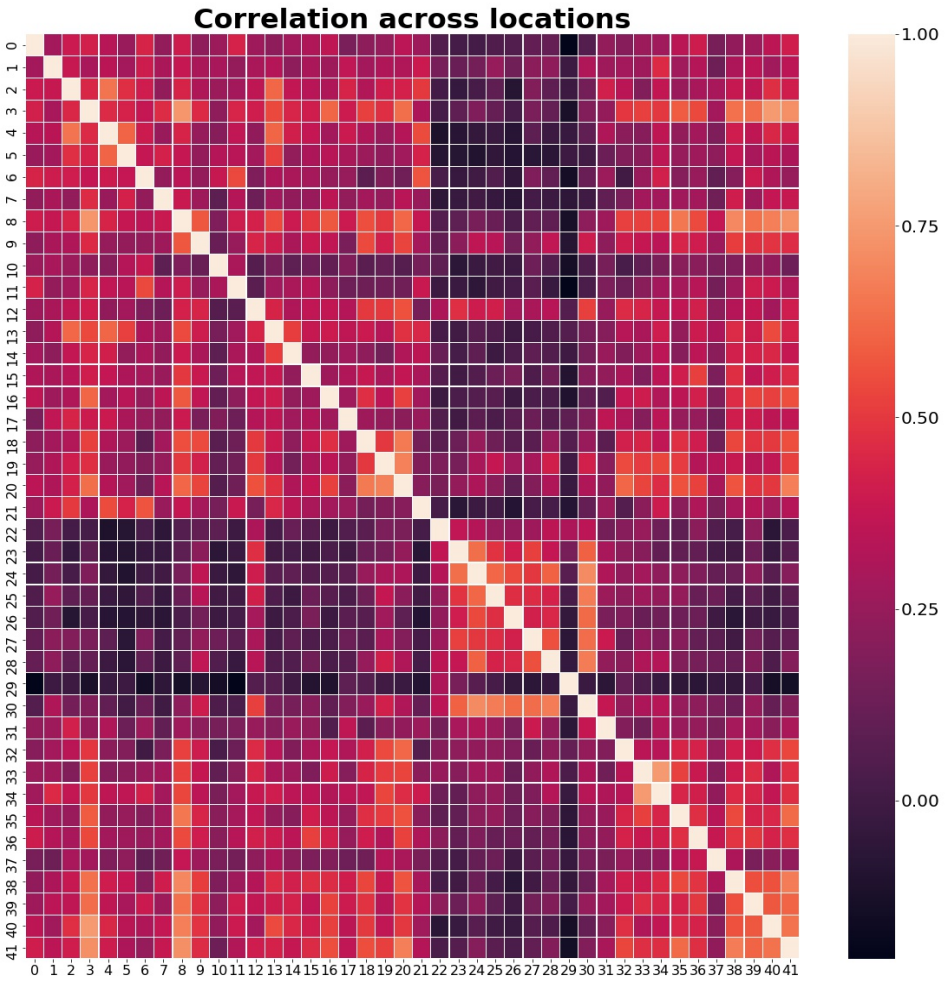

Figure 10: Spatial Correlation Heatmap on the locations with at least 50 non-zero counts. Axes represent the 42 locations

This thus provides the motivation to use the existence of spatial correlation and explore the usage of VAR model.

6.3 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 [24] and [15]. 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 [2].

6.4 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 [21].

Next, BigVAR Library in R was used to fit a VAR model with a maximum lag of 12, using time series of every location as predictors. The results for the VAR model is discussed in the later subsection.

6.4.1 Cointegration

[4] 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 [8], 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.

6.4.2 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 [11]. 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.

6.5 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 lagged endogenous variables used, s is the number of lagged 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_i are coefficient matrices for exogenous coefficient matrix for $i > 0$ and α_t are independent and identically distributed random vectors.

6.6 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 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.

6.7 Results

On the small dataset, the MSFE using the VAR model is **42.76** while the MSFE using the VARX model is **41.73**. On the full dataset, the MSFE using the VAR model is **76.72** while the MSFE using the VARX model is **75.63**.

Since the VARX includes exogenous variables and have a better result compared to the VAR model, it shows that the exogenous variables used do have a moderate amount of explanatory power. To try and improve our forecast accuracy further, the next section will introduce a 3-step modelling approach.

7 3-step Approach

In this section, we introduce an alternative 3-step approach, which combined clustering, an unsupervised learning method, along with VAR method mentioned above. 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 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 maintain a relative constant distribution as before.

Similar methods can be found in [17] and [5]

7.1 Clustering

A simple K-means 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 K-means clustering.

The first metric used would be to cluster 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.

7.1.1 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 using their geo-location coordinates (longitude and latitude). K-means clustering is performed using the euclidean distance of the locations as the distance metric.

7.1.2 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. 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. Correlation measures the strength and direction of the tendency for any 2 time series to move together.

7.1.3 Using Dynamic Time Warping as the distance metric

Finally, we explored using Dynamic Time Warping (DTW) Distance as the distance metric for K-means 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:

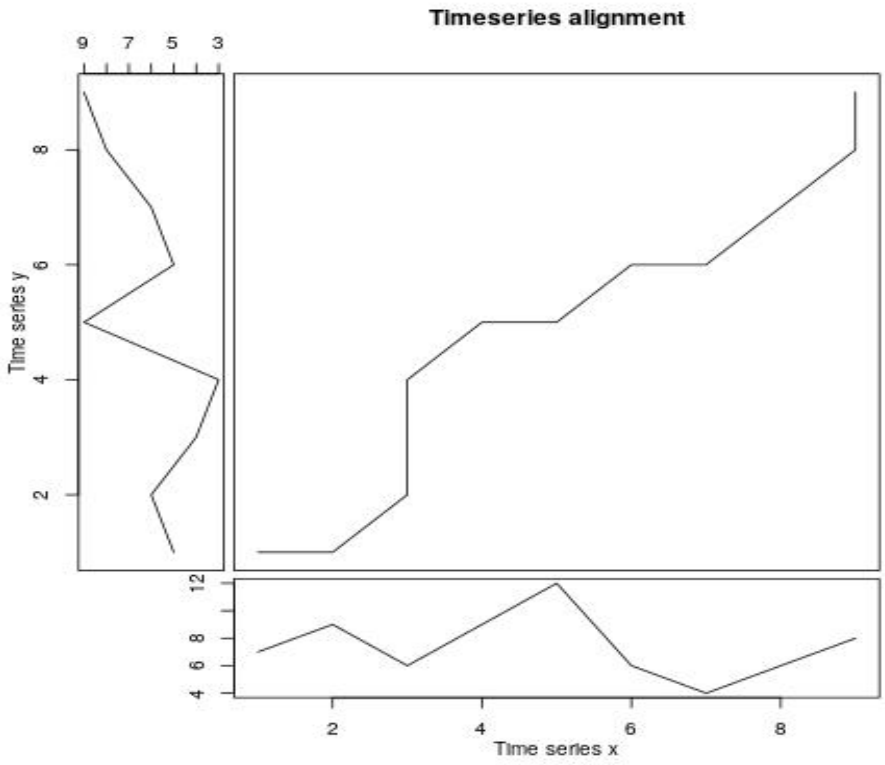

Figure 11: DTW Alignment Plot

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 [9], 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 [18]

7.2 Predicting total demand for each cluster

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

$$x \, \epsilon \, C(i)$$

$$D_{C(i)} = \sum_{l \epsilon C(i)} \sum_{j=1}^{n} x_{lj}$$

if location x belongs to Cluster i and where $D_{C(i)}$ represents the total demand (from training set) of the cluster C(i) and x_{ij} represents the training set demand at location i at time j.

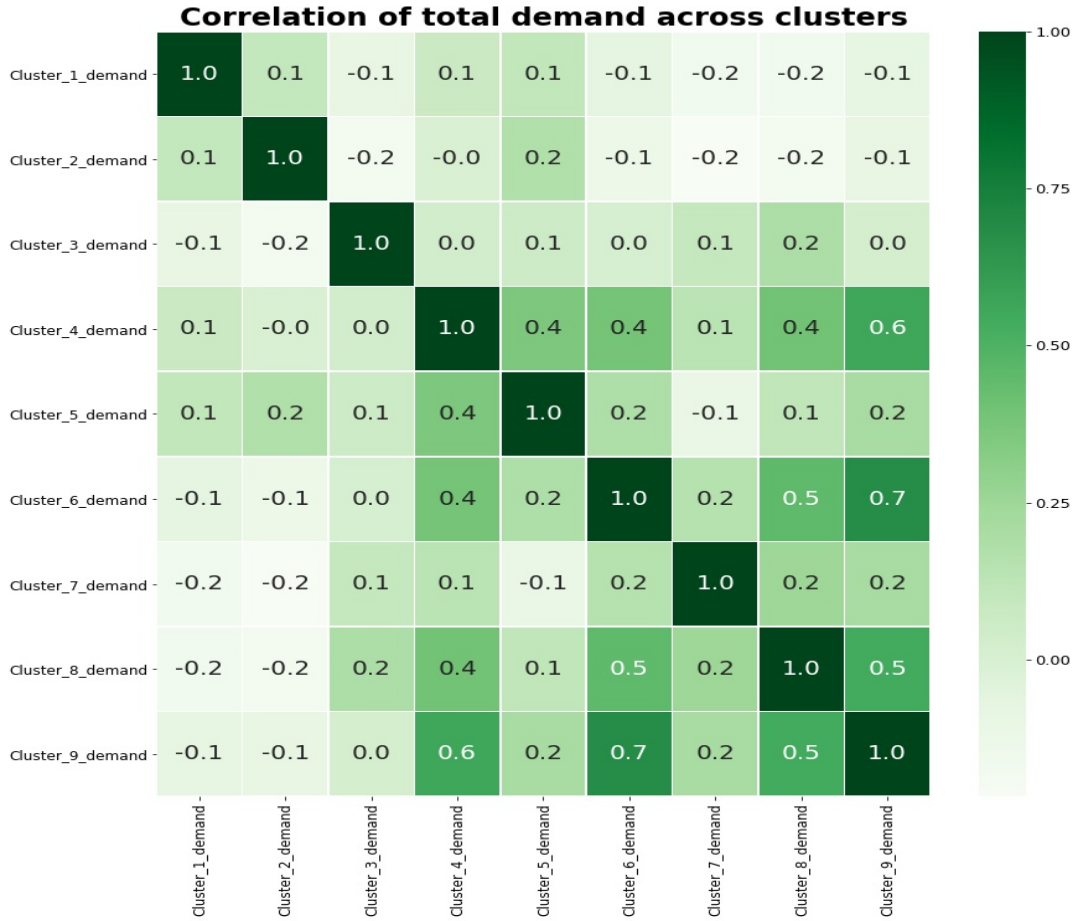

Figure 12: Correlation heatmap for 9 clusters

We can observe from the sample correlation heatmap in Figure 16 that there exist strong correlation between certain clusters (like cluster 3 and 4 and cluster 6 and 8). To exploit the spatial correlation effects, similar steps as mentioned in Section 6 was used.

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. A VAR model was then trained using the clusters as variables.

4. If we have performed differencing in step 2, we then use the VAR 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 since it does not require the aggregation of exogenous variables.

7.3 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 Cluster 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).

7.4 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 model on them and reallocate the demand using the VAR model predictions. Hence, we tried different number of clusters for the 3 different distance metric as mentioned in Section 7.1 and observe the MSFE for each of those.

	3 Clusters	**6 Clusters**	**9 Clusters**	**12 Clusters**
Clustering using euclidean distance of latitude/longitude	64.39	63.00	**60.12**	63.39
Clustering using correlation between time series as distance metric	63.45	63.97	**59.27**	63.27
Clustering using DTW as distance metric	66.39	**62.07**	65.67	62.56

Table 1: MSFE for different clustering distance metrics and number of clusters. Bolded results signifying the best result for the respective distance metric.

Based on the results, clustering the time series into 9 groups using the correlation between time series as the distance metric seems to give the best result.

7.4.1 Model Diagnostics

To test the adequacy of our model fitness, particularly our VAR model, we will plot the residuals against the fitted values and observe if there are signs of heteroscedasticity.

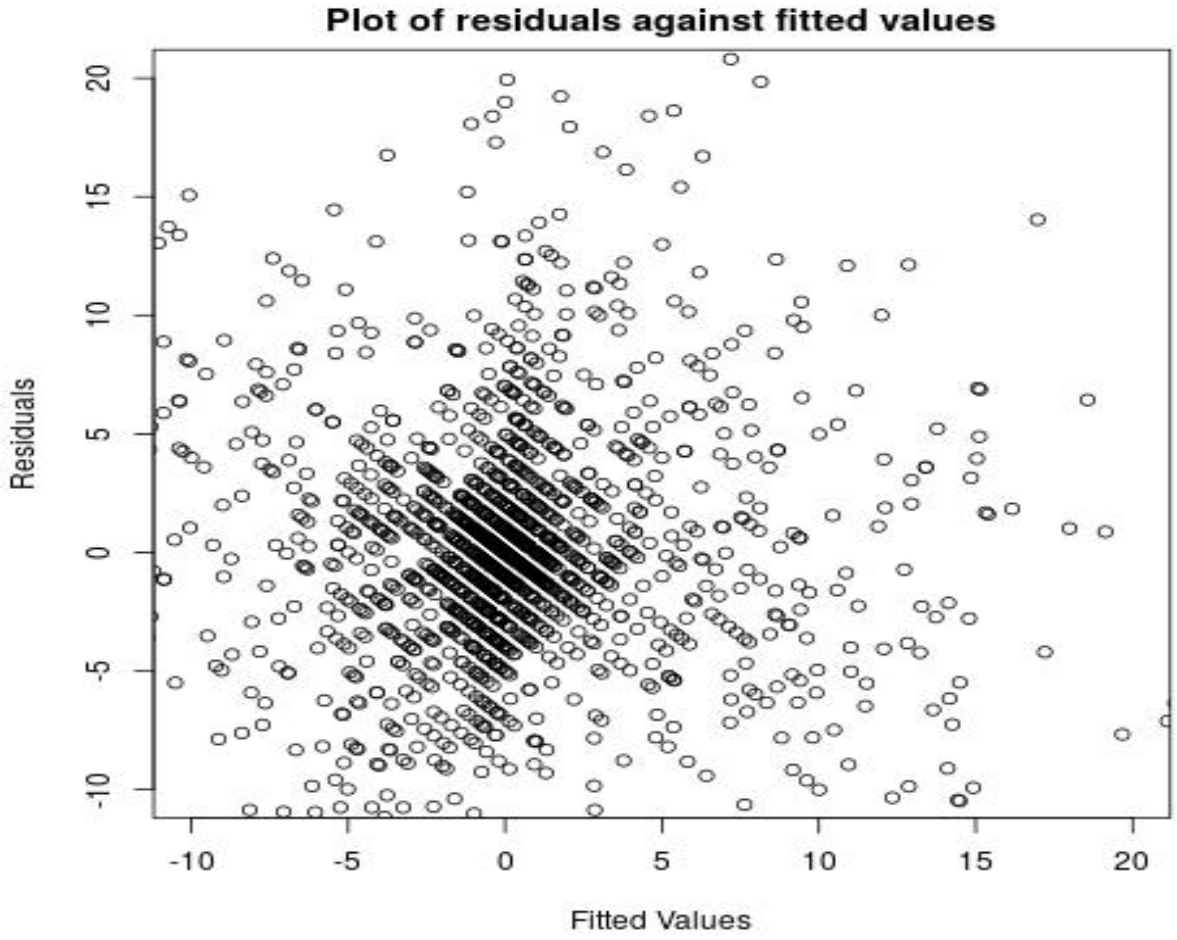

Figure 13: Plot of residuals against fitted values for VAR model

From the plot of residuals against the fitted values of the VAR model, we observe that although there seem to be a concentration of residuals around the predicted values of -5 to 5, there is no clear sign of heteroscedasticity, hence our model is reasonably adequate.

8 Results and Conclusion

The table below summarises the results of our models.

	MSFE (Locations with at least 50 non-zero counts)	**MSFE (All locations)**
ARIMA	47.60	72.48
GLM	38.16	82.42
VAR	42.76	76.72
VARX	41.73	75.63
3-step approach (Using K-means on geographical location)	40.43	60.12
3-step approach (Using K-means on correlation of time series)	39.49	**59.27**
3-step approach (Using K-means on DTW of time series)	40.10	62.07

Table 2: MSFE of the different methods used in this paper

We can see that the VAR and VARX model 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 cases, 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 on all the locations. This might be due to the 3-step approach being more reliable and reasonable. In the 3-step approach, our VAR model uses just the 9 clusters as variables to predict the total cluster demand for just 9 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. This is similar to the findings from [1], which states that a simple VAR model would not perform as well for high-dimensional data.

As for the distance metric used for clustering, using correlation also gives better results than using euclidean distance between the geo-locations, and this would imply that spatial correlation is not 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 best 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.

9 Limitations and Future Work

Our 3-step approach model display decent MSFE 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 to forecast the demand across locations. LSTM or 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, which might lead to over-differencing for some series.

Another area that could be explored further would be the method used to assign the total cluster demand to individual locations. Currently, the method simply used the historical distribution of the total sum of demand of each location over all the locations in that cluster. This would only work well if the distribution of the demand for the locations remain relatively constant over time. However, if the distribution of the locations fluctuates greatly, the current reassignment formula might not give a good result. 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.

10 Appendix

10.1 Distribution Plots of Exogenous Variables

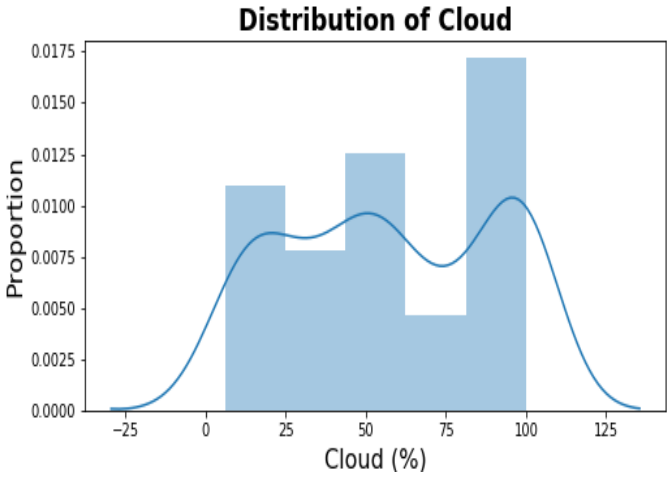

Figure 14: Distribution of Cloud

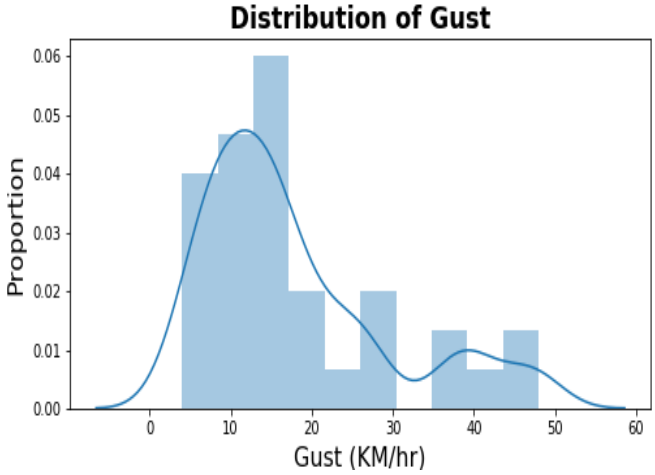

Figure 15: Distribution of Gust

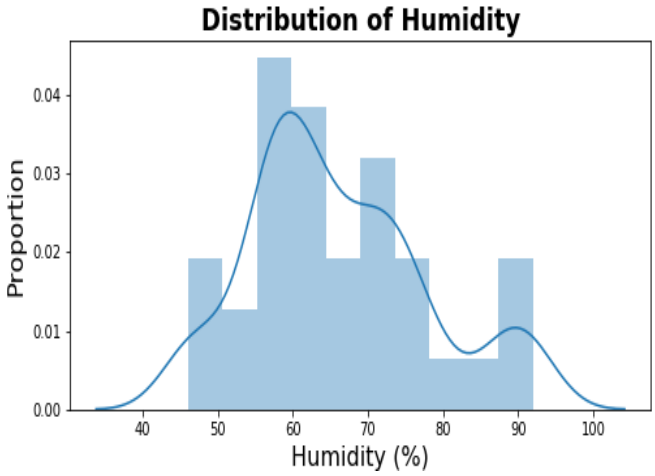

Figure 16: Distribution of Humidity

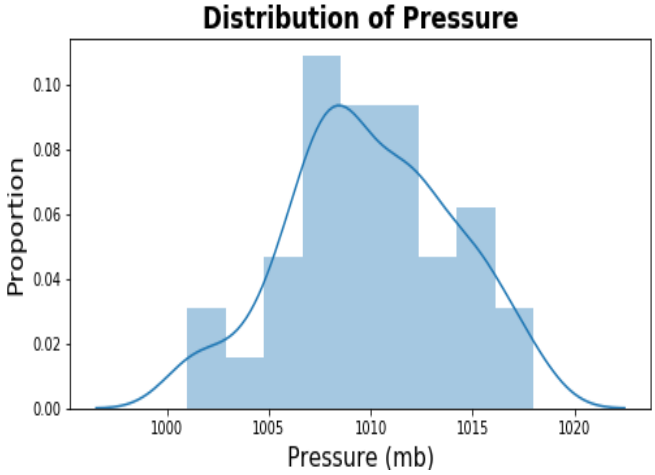

Figure 17: Distribution of Pressure

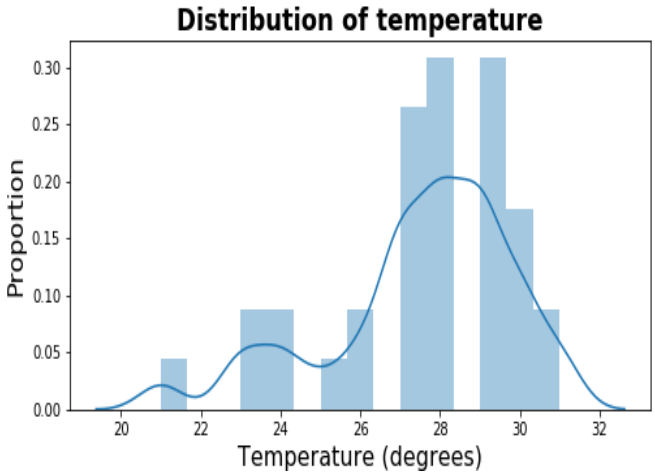

Figure 18: Distribution of Temperature

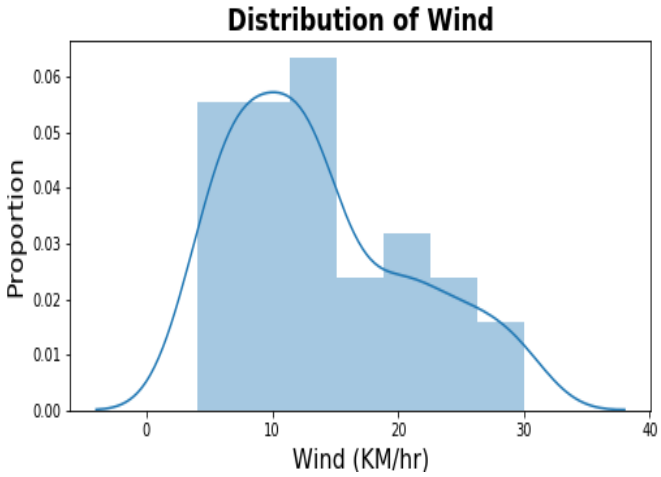

Figure 19: Distribution of Wind

10.2 Mean Demand Against Exogenous Variables

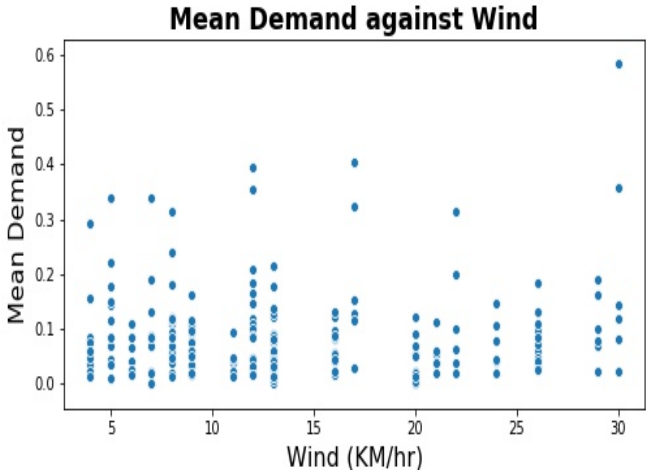

Figure 20: Mean Demand against Wind

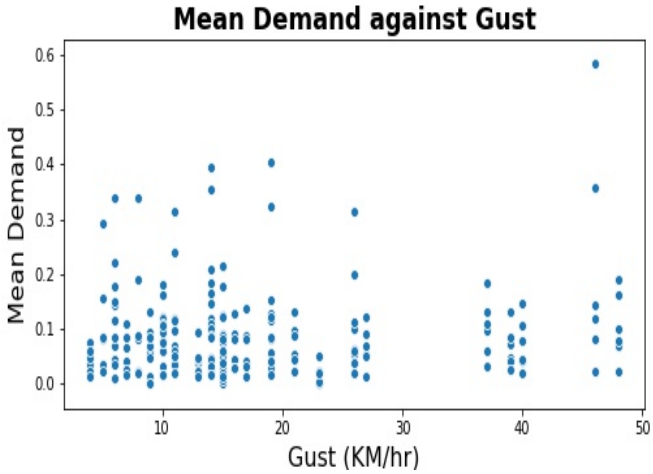

Figure 21: Mean Demand against Gust

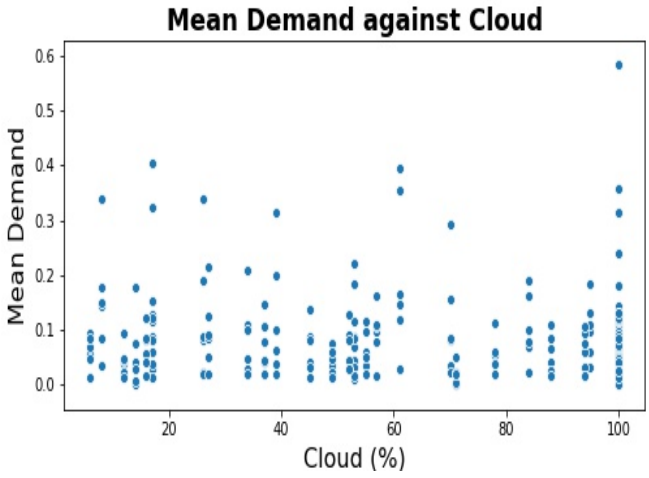

Figure 22: Mean Demand against Cloud

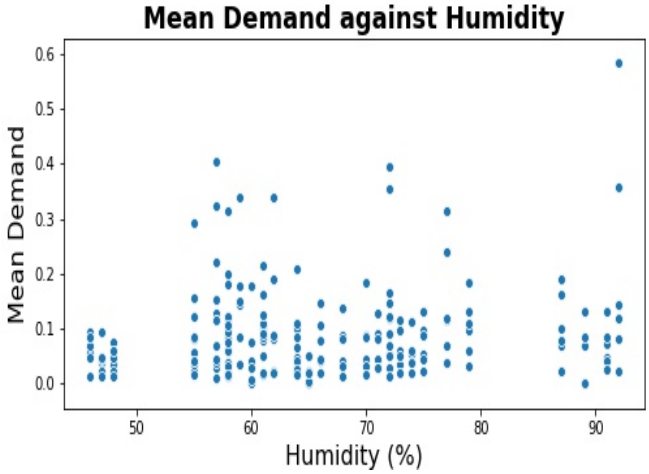

Figure 23: Mean Demand against Humidity

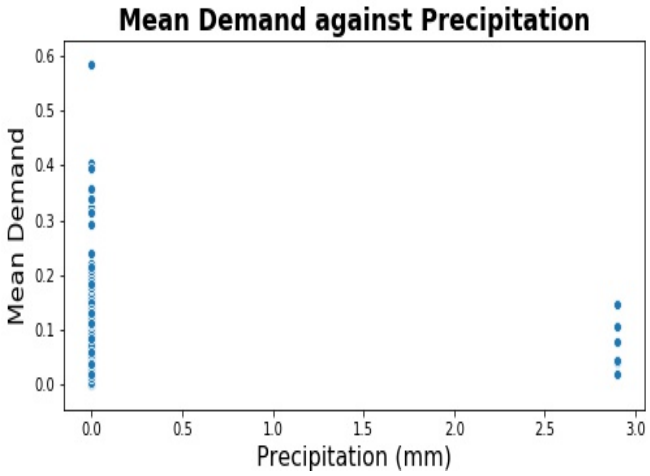

Figure 24: Mean Demand against Precipitation

References

[1] S. M. S. S. F. B. M. Abolfazl Safikhani, Camille Kamga. Spatio-temporal modeling of yellow taxi demands in new york city using generalized star models. 2017.

[2] L. . Baumhl, Eduard. Stationarity of time series and the problem of spurious regression. 2009.

[3] H. C. Bjrnland. Var models in macroeconomic research. 2000.

[4] T. G. C. Box, G. E. P. A canonical analysis of multiple time series. biometrika64(2):355365. 1977.

[5] C.-C. C. Chi-Jie Lu. A hybrid sales forecasting scheme by combining independent component analysis with k-means clustering and support vector regression. 2014.

[6] X. de Luna and M. G. Genton. Predictive spatio-temporal models for spatially sparse enviromental data. 2005.

[7] S. N. Dongjie Wang, Yan Yang. Deepstcl: A deep spatio-temporal convlstm for travel demand prediction. 2018.

[8] C. Granger. Co-integrated variables and error-correcting models, ucsd discussion paper 83-13. 1983.

[9] P. S. X. W. E. K. Hui Ding, Goce Trajcevski. Querying and mining of time series data: Experimental comparison of representations and distance measures. 2008.

[10] Z. A. H. E. M. A. L. Jamal Fattah, Latifa Ezzine. Forecasting of demand using arima model. 2018.

[11] S. Johansen. Estimation and hypothesis testing of cointegration vectors in gaussian vector autoregressive models. 1991.

[12] M. N. Marina Knight and G. Nason. Modelling, detrending and decorrelation of network time series. 2016.

[13] G. P. N. Matthew A. Nunes, Marina I. Knight. Modelling and prediction of time series arising on a graph. 2015.

[14] D. Molenaar and M. Bolsinova. A heteroscedastic generalized linear model with a nonnormal speed factor for responses and response times. 2017.

[15] R. Mushtaq. Augmented dickey fuller test. 2011.

[16] W. R. Nelder JA. generalized linear models. journal of the royal statistical society a, 135(3), 370384. 1972.

[17] M. A. Paul W.Murraya, Bruno Agardb. Forecasting supply chain demand by clustering customers. 2015.

[18] P. Senin. Dynamic time warping algorithm review. 2017.

[19] Z.-J. M. S. Sheng Liu, Long He. On-time last mile delivery: Order assignment with travel time predictors. 2018.

[20] R. F. Tobias Liboschik, Konstantinos Fokianos. tscount: An r package for analysis of count time series following generalized linear models. 2017.

[21] R. S. Tsay. Multivariate time series analysis with r and financial applications. 2014.

[22] J. B. William Nicholson, David Matteson. Bigvar: Tools for modeling sparse high-dimensional multivariate time series. 2017.

[23] L. A. Z. Asha Farhath, B. Arputhamary. A survey on arima forecasting using time series model. 2016.

[24] P. C. P. Zhijie X. An adf coefficient test for a unit root in arma models of unknown order with empirical applications to the us economy. 1998.