Recruit Restaurant Visitor Forecasting

ISDS 577

By

Group 1

Abhilash Urs

Aishwarya Murali

Jaymin Joshi

Kedar Kulkarni

Sriram Ramanujam

CALIFORNIA STATE UNIVERSITY
FULLERTON
MIHAYLO COLLEGE
OF BUSINESS AND ECONOMICS

Table of Contents

Introduction:

Recruit Holding is a global company which provides human resource services. The company has achieved great success in the segments of human resources technology and staffing. Recruit has accessibility to handle the reservations of all the restaurants in Japan. With the data available in their system, Recruit has always managed staff, efficiently dealt with the inventory management and scheduled the shifts.

The reservation data collected by Recruit has three sources: AirREGI, Hot Pepper Gourmet, and Restaurant Boards. The sources of the Recruit reservation data have shown in Exhibit 1.

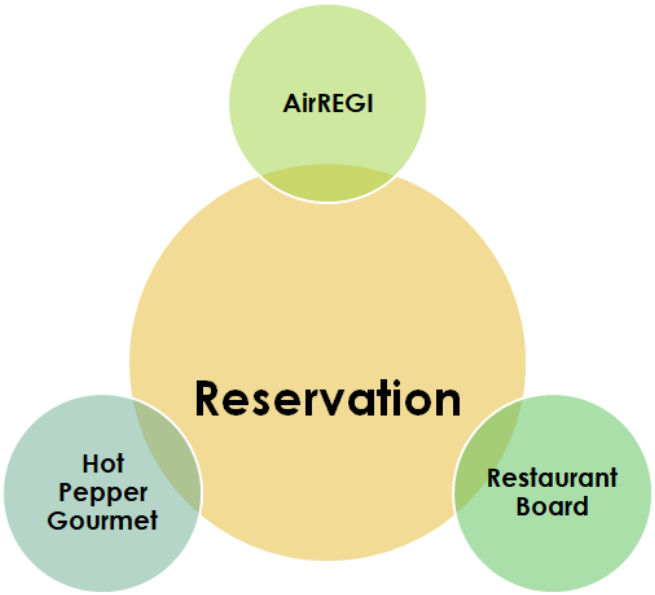
AirREGI is a cash register and reservation control system. AirREGI uses POS app for everyday operation at restaurants from table management, order, payments to analysis. A point of sale (POS) is a combination of software and hardware device which is used to record transactions in the restaurant.

Hot Pepper Gourmet is an online platform to search restaurants and make reservations online. The online platform allows to record ratings, read reviews from customers, gives suggestions for restaurants and restaurant coupons for next visit.

Restaurant boards are manually collected reservation data and records on the local computers of the restaurant. It is a reservation log management

technique for walk-ins. Many restaurants have switched to new technologies and platforms but there are very few restaurants follows the traditional data recording systems.

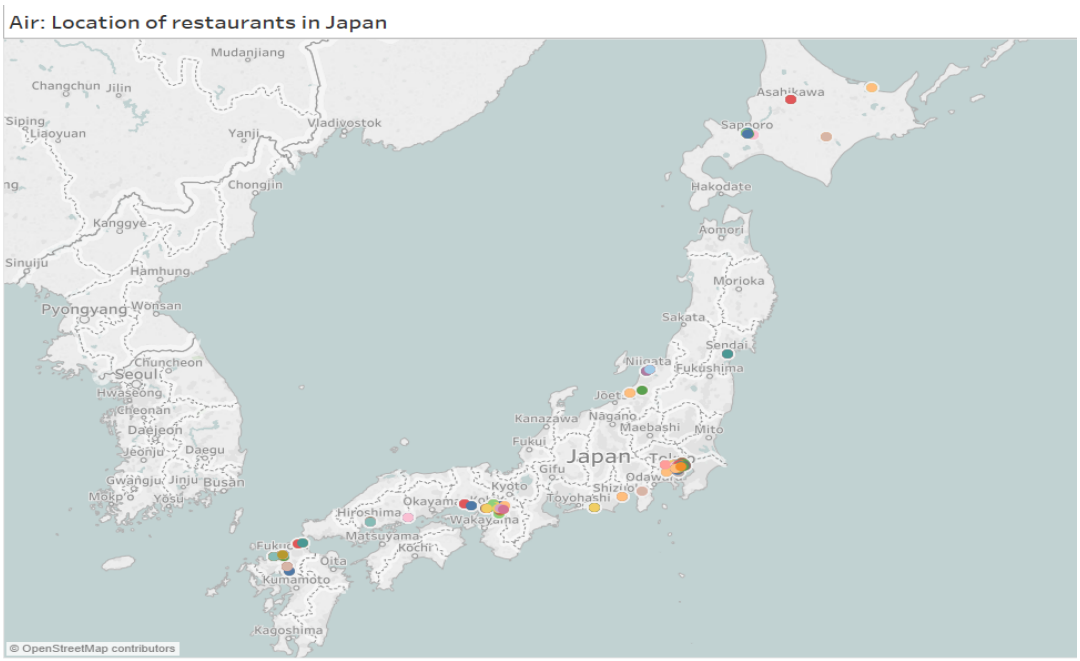
Exhibit 2: Reservation data sources for Recruit



The company is predicting the number of visitors for their restaurants in Japan for the period 23rd April 2016 to 31st May 2017. The company possess data of 825 restaurants registered under Air system and 4690 restaurants under HPG system. Recruit Holdings has access to the reservation data for all these restaurants. Out of all these restaurants 150 restaurants allows to reserve from both the systems. Recruit holding is looking to manage employees, resources, tables for each restaurant in the

future dates. The team is working on creating models which will help Recruit Holding to forecast a total number of visitors to provide efficient service and creating an enjoyable dining experience for their customers. Exhibit 3 shows major locations of Air systems restaurants in Japan.

Exhibit 3: Location of Air restaurants in Japan



The management of the Recruit are interested in knowing the insights by forecasting on the number of visitors in a day; also, they want to know the busiest day, week and month for the restaurants. With the knowledge of the information, the management can efficiently work on staffing, scheduling and inventory management. Many times, people reserve the table for the evening and could not show up for the reservation. This problem can be dealt by predicting the conversions of reservations to

actual visits using the past data and experience from the restaurant management. The company has plans to expand their business in Japan, to achieve this the data analysis team is considering the customer behavior with the choice of cuisine and location of restaurants. The analytical team and management are working hard to achieve these milestones in the given period.

Attribute/ Variables of Datasets:

The project has 7 datasets. The datasets are related to each other.

following attributes are available in each dataset:

air_reserve: This file contains reservations made in the air system

Variable	**Data type**	**Description**
air_store_id	alphanumeric	Restaurant's id in the air system
visit_datetime	date	Time in the future where the visit will occur (MM-DD-YYYY HH:MM)
reserve_datetime	date	Time when the reservation was created (MM-DD-YYYY HH:MM)
reserve_visitors	integer	Number of visitors for that reservation

air_store_id	visit_datetime	reserve_datetime	reserve_visitors
air_877f79706adbfb06	1/1/2016 19:00	1/1/2016 16:00	1
air_db4b38ebe7a7ceff	1/1/2016 19:00	1/1/2016 19:00	3
air_db4b38ebe7a7ceff	1/1/2016 19:00	1/1/2016 19:00	6
air_877f79706adbfb06	1/1/2016 20:00	1/1/2016 16:00	2
air_db80363d35f10926	1/1/2016 20:00	1/1/2016 1:00	5
air_db80363d35f10926	1/2/2016 1:00	1/1/2016 16:00	2
air_db80363d35f10926	1/2/2016 1:00	1/1/2016 15:00	4
air_3bb99a1fe0583897	1/2/2016 16:00	1/2/2016 14:00	2
air_3bb99a1fe0583897	1/2/2016 16:00	1/1/2016 20:00	2

hpg_reserve: This file contains reservations made in the hpg system

Variable	Data type	Description
hpg_store_id	alphanumeric	Restaurant's id in the hpg system
visit_datetime	date	Time in the future where the visit will occur (MM-DD-YYYY HH:MM)
reserve_datetime	date	Time when the reservation was created (MM-DD-YYYY HH:MM)
reserve_visitors	integer	Number of visitors for that reservation

hpg_store_id	visit_datetime	reserve_datetime	reserve_visitor
hpg_c63f6f42e088e50f	1/1/2016 11:00	1/1/2016 9:00	1
hpg_dac72789163a3f47	1/1/2016 13:00	1/1/2016 6:00	3
hpg_c8e24dcf51ca1eb5	1/1/2016 16:00	1/1/2016 14:00	2
hpg_24bb207e5fd49d4a	1/1/2016 17:00	1/1/2016 11:00	5
hpg_25291c542ebb3bc2	1/1/2016 17:00	1/1/2016 3:00	13
hpg_28bdf7a336ec6a7b	1/1/2016 17:00	1/1/2016 15:00	2
hpg_2a01a042bca04ad9	1/1/2016 17:00	1/1/2016 17:00	2
hpg_2a84dd9f4c140b82	1/1/2016 17:00	1/1/2016 15:00	2
hpg_2ad179871696901f	1/1/2016 17:00	1/1/2016 13:00	2

air_store_info: This file contains information about select air restaurants

Variable	Data type	Description
air_store_id	alphanumeric	Restaurant's id in the air system
air_genre_name	string	Genre of the restaurant
air_area_name	string	Area of the restaurant
Latitude	integer	Latitude of the area to which the restaurant belongs
Longitude	integer	Longitude of the area to which the restaurant belongs

air_store_id	air_genre_name	air_area_name	latitude	longitude
air_0f0cdeee6c9bf3d7	Italian/French	HyÅgo-ken KÅbe-shi KumoidÅri	34.6951242	135.1978525
air_7cc17a324ae5c7dc	Italian/French	HyÅgo-ken KÅbe-shi KumoidÅri	34.6951242	135.1978525
air_fee8dcf4d619598e	Italian/French	HyÅgo-ken KÅbe-shi KumoidÅri	34.6951242	135.1978525
air_a17f0778617c76e2	Italian/French	HyÅgo-ken KÅbe-shi KumoidÅri	34.6951242	135.1978525
air_83db5aff8f50478e	Italian/French	TÅkyÅ-to Minato-ku ShibakÅen	35.6580681	139.7515992
air_99c3eae84130c1cb	Italian/French	TÅkyÅ-to Minato-ku ShibakÅen	35.6580681	139.7515992
air_f183a514cb8ff4fa	Italian/French	TÅkyÅ-to Minato-ku ShibakÅen	35.6580681	139.7515992
air_6b9fa44a9cf504a1	Italian/French	TÅkyÅ-to Minato-ku ShibakÅen	35.6580681	139.7515992
air_0919d54f0c9a24b8	Italian/French	TÅkyÅ-to Minato-ku ShibakÅen	35.6580681	139.7515992

hpg_store_info: This file contains information about select hpg restaurants

Variable	**Data type**	**Description**
hpg_store_id	alphanumeric	Restaurant's id in the hpg system
hpg_genre_name	string	Genre of the restaurant
hpg_area_name	string	Area of the restaurant
Latitude	integer	Latitude of the area to which the restaurant belongs
Longitude	integer	Longitude of the area to which the restaurant belongs

hpg_store_id	hpg_genre_name	hpg_area_name	latitude	longitude
hpg_6622b62385aec8bf	Japanese style	TÅkyÅ-to Setagaya-ku TaishidÅ	35.64367466	139.6682209
hpg_e9e068dd49c5fa00	Japanese style	TÅkyÅ-to Setagaya-ku TaishidÅ	35.64367466	139.6682209
hpg_2976f7acb4b3a3bc	Japanese style	TÅkyÅ-to Setagaya-ku TaishidÅ	35.64367466	139.6682209
hpg_e51a522e098f024c	Japanese style	TÅkyÅ-to Setagaya-ku TaishidÅ	35.64367466	139.6682209
hpg_e3d0e1519894f275	Japanese style	TÅkyÅ-to Setagaya-ku TaishidÅ	35.64367466	139.6682209
hpg_530cd91db13b938e	Japanese style	TÅkyÅ-to Setagaya-ku TaishidÅ	35.64367466	139.6682209
hpg_02457b318e186fa4	Japanese style	TÅkyÅ-to Setagaya-ku TaishidÅ	35.64367466	139.6682209
hpg_0cb3c2c490020a29	Japanese style	TÅkyÅ-to Setagaya-ku TaishidÅ	35.64367466	139.6682209
hpg_3efe9b08c887fe9a	Japanese style	TÅkyÅ-to Setagaya-ku TaishidÅ	35.64367466	139.6682209

air_visit_data: This file contains historical visit data for the air restaurants

Variable	Data type	Description
air_store_id	alphanumeric	Restaurant's id in the air system
visit_date	date	Date of the Visit
Visitors	integer	Number of visitors

air_store_id	visit_date	visitor
air_ba937bf13d40fb24	1/13/2016	25
air_ba937bf13d40fb24	1/14/2016	32
air_ba937bf13d40fb24	1/15/2016	29
air_ba937bf13d40fb24	1/16/2016	22
air_ba937bf13d40fb24	1/18/2016	6
air_ba937bf13d40fb24	1/19/2016	9
air_ba937bf13d40fb24	1/20/2016	31
air_ba937bf13d40fb24	1/21/2016	21
air_ba937bf13d40fb24	1/22/2016	18

store_id_relation: This file contains information about the join of selected restaurants that have both the air and hpg system

Variable	Data type	Description
air_store_id	alphanumeric	Restaurant's id in the air system
hpg_store_id	alphanumeric	Restaurant's id in the hpg system

air_store_id	hpg_store_id
air_63b13c56b7201bd9	hpg_4bc649e72e2a239a
air_a24bf50c3e90d583	hpg_c34b496d0305a809
air_c7f78b4f3cba33ff	hpg_cd8ae0d9bbd58ff9
air_947eb2cae4f3e8f2	hpg_de24ea49dc25d6b8
air_965b2e0cf4119003	hpg_653238a84804d8e7
air_a38f25e3399d1b25	hpg_50378da9ffb9b6cd
air_3c938075889fc059	hpg_349b1b92f98b175e
air_68301bcb11e2f389	hpg_2c09f3abb2220659
air_5f6fa1b897fe80d5	hpg_40aff6385800ebb1

Date_info: This file gives information about the calendar dates in the dataset

Variable	Data type	Description
calender_date	date	Date in the dataset (MM-DD-YYYY)
day_of_week	string	Day of the week
holiday_flg	boolean	Flag for holiday in Japan (1: Holiday, 0: No holiday)

calendar_date	day_of_week	holiday_flg
1/1/2016	Friday	1
1/2/2016	Saturday	1
1/3/2016	Sunday	1
1/4/2016	Monday	0
1/5/2016	Tuesday	0
1/6/2016	Wednesday	0
1/7/2016	Thursday	0
1/8/2016	Friday	0
1/9/2016	Saturday	0

Data Cleaning:

The main task is to merge the various data sets and obtain data integrity to answer research questions.

We had many different files where the data was scattered. So, the first step that we needed to do is data cleaning and merging. We merged all different data files and made a combined data file with all useful variables. We decided to proceed with MS SQL for data cleaning.

First, we needed to convert the hourly data of air_reserve to daily basis, so we made a file name called air_data where we made this change. We did the same for hpg, converting hourly data into daily data. We implemented the idea that is:

air_visit_data = Air reserve data + Hpg reserve data + Walk-ins.

Now, we need to sort the data from store_id_relation, so from store_id_relation we made a new data file name hpg_storeid with hpg_store_id. We matched the relative hpg_store_id with air_store_id from store_id_relation table and inserted the values in a new file called air_data. The, we combined air_data which consists of daily records of air_reserve and airdata which consists of related entries of air_store_id which contains hpg data and stored it in. The file that we obtained has air_store_id, visit_date, visitors_1, visitors_2 where visitors_1 had visitors of air restaurants and visitors_2 had visitors from hpg restaurants. Visitors_2 had many null entries as we only took entries which were relevant with hpg restaurants, so we converted those null entries into 0 and the added visitors_1 and visitors_2 and made a combined attribute as visitors.

Now we need to find the Genre and the location, so we made a new file called distinct_store_id which contains the unique air store id from data file. Then we checked the tables distinct_store_id and air_store_info and found out that both the table contain 829 records which are related to each other. So, we decided to join data and air_store_info table to fetch air_store_id, area_name, longitude, latitude, visit date, visitors visited, and visitors reserved. The file name air_combined contains all the information.

Below is the table which was obtained by executing sp_columns air_combined and it contains columns name, datatypes, and type name of the data file.

TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION	LENGTH
air_combined	air_store_id	12	varchar	50	50
air_combined	air_genre_name	12	varchar	50	50
air_combined	air_area_name	12	varchar	50	50
air_combined	latitude	12	varchar	50	50
air_combined	longitude	12	varchar	50	50
air_combined	visit_date	-9	date	10	20
air_combined	visitors_visited	4	int	10	4
air_combined	visitors_reserved	4	int	10	4

Data Pre-Processing (Time Series):

Our main objective is to forecast the number of visitors for a restaurant. Forecasting is the process of making predictions of the future based on historic data and time is an important factor. Forecasting is generally performed on a time series data. A time series is a collection of data over a unit of time (i.e. historical data).

Our cleaned dataset is does not contain any missing values and the granularity in the dataset is set to daily. We have "visit_date" has one of the attributes in the dataset and this attribute is used to build a time series data on which a forecasting model can be built. To build a time series data, we should:

- Sort the data

- Fill missing dates
- Handle missing values
- Daily Data with weekly frequency
- Time Series Objects
- Data Transformation
- Holdout (Testing set)

Sort the Data:

We sorted the data for each restaurant by date in R. The time series data starts from the earliest date (1st January 2016) and goes till the latest date which is 22nd April 2017 in our dataset. The start date is different for each restaurant since most of the restaurants opened later or we don't have the earlier dates in our dataset. But the end date is the same (22nd April 2017) for all restaurants.

Fill Missing Dates:

Any time series data should be continuous in its unit of time. We are building a daily time series data, so we checked for missing dates in the sorted data of each restaurant. And if there are any missing dates, we filled them in each time series data respective to each restaurant. After filling the missed dates, the time series data we have, is continuous.

Handle Missing Values:

Now that we filled in the missing dates in the time series data, we have introduced missing values (NA) for the number of visitors (visitors_visited) attribute. We should handle the missing values in our time series and we know from our source (Kaggle) that the missing dates in the dataset are the days on which the restaurants are closed. There are no visitors and no reservations for the closed days. and so we cannot impute the values for our continuous variables of no. of visitors (visitors_visited) and no. of reservations (visitors_reserved).

We introduce a new variable here called "closed_days", which act as an event in the time series data, marking the closed days of the restaurants as "1" and open days as "0".

Daily Data with weekly frequency:

Our dataset is aggregated into a daily data to better suit our objectives and for time series modeling. From our data analysis, we found that the data has a weekly frequency, which is that the patterns of visitors and reservation have a week of the day effect. For e.g., Friday, Saturday and Sunday seem to have more visitors than the rest of the week.

To capture the seasonal patterns on a weekly frequency, we introduced weekly seasonal dummy variables:

1. mon_thurs ("1" (True) if the day is Monday to Thursday)

2. friday ("1" (True), when the day is Friday)
3. saturday ("1" (True), when the day is Saturday)

For these categorical weekly seasonal variables, Sunday is taken as the reference. So, when all three variables are "0", it means that it is Sunday. After athis, the final dataset is fi

Data Transformation:

The key variable that is to be forecasted is the "visitors_visited" and the supporting independent variables are "visitors_reserves", "closed_days", "mon_thurs", "friday", "saturday".

The variables "visitors_visited" and "visitors" are changing numerical values that are not in the same scale. So, we took natural logarithm of both and stored them as separate attributes - "log_visitors" and "log_reserves".

Time Series Objects:

The dependent variable or the key variable that is to be forecasted is the "log_visitors" which is the "Y". And the other useable independent variables are "log_reserves", "closed_days", "mon_thurs", "friday" and "saturday".

To forecast, we must create time series objects from our dataset and this can be done in R by using the ts() function in "forecast" package.

We have 829 restaurants in our dataset, so when trying to forecast the "log_visitors" for each restaurant, we will have 829 unique time series.

Now for each restaurant, we created one time series object containing the "Y" (log_visitors). And another time series object containing other useful variables "log_reserves", "closed_day, "mon_thurs", "friday" and "saturday", which are available and can be used for model building (satisfying the conditions shown the below code).

```
temp_y <- temp$log_visitors
#temp_x - independent variables:
if (length(which(temp$visitors_reserved != 0)) < 40 | sum(temp$closed_days) == 0){
  if (length(which(temp$visitors_reserved != 0)) < 40 & sum(temp$closed_days) == 0) {
    temp_x <- temp[,c(13:15)]
  }else {
      if (length(which(temp$visitors_reserved != 0)) < 40) {
        temp_x <- temp[,c(12:15)]
      }else {
        temp_x <- temp[,c(11,13:15)]
      }
  }
}else {
  temp_x <- temp[,c(11:15)]
}

#time series objects
temp_ts_y <- ts(temp_y, frequency = 7)
temp_ts_x <- ts(temp_x, frequency = 7)
```

Holdout (Testing Set):

To build and validate a forecast model, we need a testing set from our data that is withheld from training dataset used in the model. Our main objective (from Kaggle problem) is to forecast the number of visitors from 23nd April 2017 till 31th May 2017, (i.e. 39 days) and this is our forecast horizon. In forecasting, we take the testing set equal to the forecast horizon by going 39 days in the time series, this is called the holdout period (our testing test), in which we will be validating our forecasting model.

Our training and testing set time series are created by the following code snippet in R.

```
#building the training and testing test
dtrain <- subset(temp_ts_y, end = length(temp_ts_y)-39)
dtrain_x <- subset(temp_ts_x, end = dim(temp_ts_x)[1]-39 )
#dtrain_x <- subset(ts_x, end = length(ts_dat)-39)
dtest <- subset(temp_ts_y, start = length(temp_ts_y)-38)
dtest_x <- subset(temp_ts_x, start = dim(temp_ts_x)[1]-38)
```

Data Modeling for Time Series with R:

Forecasting Models – Model Building:

To begin our data modeling with forecasting in R, let's look few of the forecasting techniques that we are considering for this project:

Exponential Smoothing Family of Models:

The conceptual view of exponential smoothing of a time series is:

Times Series = Pattern (or) Signal + Noise

Pattern = Level + Trend + Seasonality + Event

There are many models in exponential smoothing family and each of them perform different smoothing techniques (like on level, trend and seasonality) and captures the patterns. So, each model performs well on different time series. Some of the models are,

- Simple Moving Average
- Simple Exponential Smoothing

- Damped Exponential Smoothing
- Holt's Exponential Smoothing (Level & Trend)
- Winter Seasonal Models (Additive & Multiplicative)
- Event Adjustment Models

For exponential smoothing models, we will be using only the "Y" time series named "dtrain" to forecast. In R, the exponential model is run from the following code shown below,

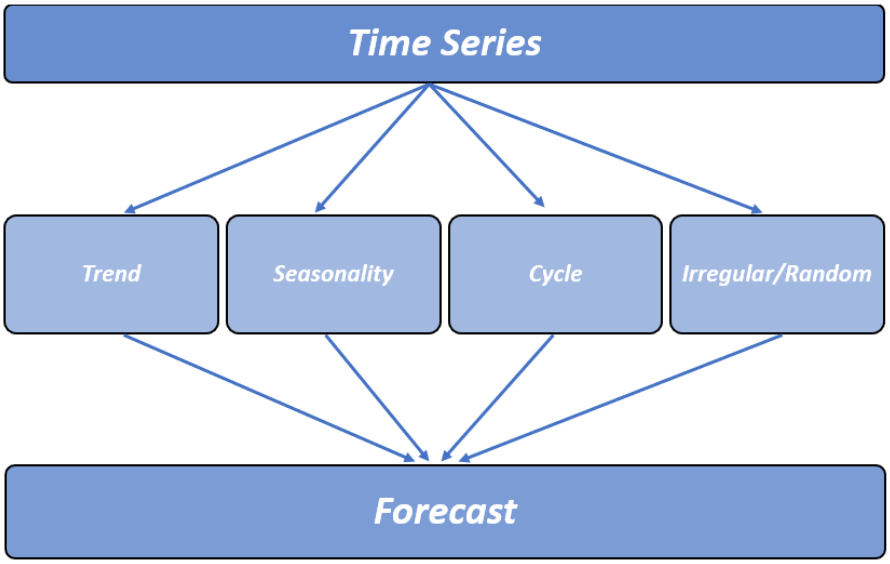
```
# 1) Exponential Smoothing Method:
mod_01 <- ets(dtrain, allow.multiplicative.trend = TRUE)
f_cast_01 <- forecast(mod_01, h = 39)
```

In "mod_01", the exponential smoothing model built for the time series is stored and it is forecasted for 39 days (h = forecast horizon). The forecasted results are stored in "f_cast_01".

Time Series Decomposition:

The time series decomposition method decomposes the given time series and helps us to see the components present in it better. It first decomposes the time series and then reconstructs while forecasting as shown Exhibit 4.

Exhibit 4: Time Series decomposition model



For time series decomposition, we will be using only the "Y" time series named "dtrain" to forecast. In R, the time series decomposition is executed from the following code snippet,

```
# 2) Time Series Decomposition:
mod_02 <- stl(dtrain, s.window = 7)
f_cast_02 <- forecast(mod_02, h = 39)
```

In "mod_02", the model for time series decomposition is fitted and stored for the given times series. The forecast for the testing set (h = 39 days) is done and is stored in "f_cast_02".

Multiple Regression:

When multiple independent variables provide information about the predictor variable, we can use regression model which can be represented by,

$$Y = \beta 0 + \beta 1\, X1 + \beta 2\, X2 + \epsilon$$

For multiple regression, time series objects were not accepted by the mlr() function, so we created data frames containing "Y" and "X" variables and split them into training and testing sets. So here, we will be using the "Y" data named "l_train" and "X" data variables names in "df_temp" data frame and "l_test_x" to forecast. In R, the multi-linear regression is run by the following code snippet,

```
mod_03 <- lm(l_train ~ log_reserves + closed_days + mon_thurs + friday + saturday, data = df_temp)

f_cast_03 <- predict(mod_03, l_test_x)
```

Here, the independent variables (log_reserves, closed_days, mon_thurs, friday, saturday) are acting as predictors in building the multi linear model which is used to predict the log_visitors (dependent variable). The model

is stored in "mod_03" and the forecast for the testing set is stored in "f_cast_03".

ARIMA:

ARIMA is a widely used statistical forecasting method, it uses both

- Auto Regressive Method (AR) - uses lagged variables
- Moving Average Method (MA) - uses lagged errors.

The generic model looks like this,

$$\hat{y}_t = \mu + \phi_1 y_{t-1} + \ldots + \phi_p y_{t-p} + e_t + \theta_1 e_{t-1} + \ldots + \theta_q e_{t-q}$$

Both are integrated to give ARIMA. The forecasting is done in a reverse here, where the errors are feed into the Box Jenkins (ARIMA) model and the time series is given as output. The model after analyzing the time series, uses

- Lagged y variable (like for non-seasonal -> Y(t-1), Y(t-2), etc. and seasonal (say weekly) -> Y(t-7), Y(t-14), etc.)
- Lagged errors like e(t-1), e(t-2), etc. (for non-seasonal) and e(t-7), e(t-14), etc. (for seasonal data (say weekly))

For ARIMA model, we will using the "Y" time series named "dtrain" to forecast. In R, the ARIMA model is run from the following code snippet,

```
# 4) ARIMA Model
mod_04 <- auto.arima(dtrain, start.p = 0, start.q = 0, start.P = 0, start.Q = 0, ic = "bic", test = "kpss")
f_cast_04 <- forecast(mod_04, h = 39)
```

The model built is stored in "mod_04" and the forecast results for the testing set of 39 days (forecast horizon) are stored in "f_cast_04".

Dynamic Regression

Dynamic regression is a forecasting technique using both regression and ARIMA models. The general equation will look like this,

$$\hat{y}_t = \beta 0 + \beta 1\, X1 + \beta 2\, X2 + \phi_1 y_{t-1} + \ldots + \phi_p y_{t-p} + e_t + \theta_1 e_{t-1} + \ldots + \theta_q e_{t-q}$$

It is a kind of multiple regression with independent variables and also ARIMA variables.

For dynamic regression, we will be using the "Y" time series named "dtrain" and the external independent variables in "X" time series named "dtrain_x" & "dtest_x" to forecast. In R, dynamic regression is called regression with ARIMA and is implemented by the following code snippet,

```
# 5) Regression with ARIMA errors ( Dynamic Regression)
mod_05 <- auto.arima(dtrain, start.p = 0, start.q = 0, start.P = 0, start.Q = 0, ic = "bic", test = "kpss", xreg= dtrain_x)
f_cast_05 <- forecast(mod_05, h = 39, xreg = dtest_x)
```

The dynamic regression model built in R is fitted with the training and is stored in the variable "mod_05". The forecasts calculated for the horizon are stored in "f_cast_05".

Model Selection Algorithm:

Each of these models performs and captures specific patterns in the time series, so one model can perform well on a time series and at the same time not so good on a different time series. This is because the time series have several patterns and each one may need a different model to forecast.

A data modeling and model selection algorithms was designed such that all the modeling is carried out for all the 829 restaurants' time series and the results are evaluated. And the model with the lowest testing set RMSLE - Root Mean Squared Log Error (Kaggle's primary performance measure for the problem) is selected as the best model was for that time series. This is called "accuracy of the model" and it is the RMSLE of the testing dataset.

R^2 - the predictive power of the model is the second criteria in our model selection algorithm. We also calculate the both,

- Train set R^2 - to access the fit of the model.
- Testing set R^2 - to check our model's performance on unseen data.

When the accuracy is good (less RMSLE), then it implies that the model was good at predicting the results. So, Lower the RMSLE for a model, the R^2 of that model will be good.

Ljung-Box (LB) Test – Health of the model is the third optional criteria in our model selection algorithm, but this requires manual analysis of the

Error ACF (Auto Correlation Function) lags. Normal ACF is a function describing the correlation between the "Y" time series at different lagged variables. Error ACF is the same but it is done for the errors of the model. LB test determines whether a time series has "serial correlation". Serial correlation is means that the errors at different time lags are correlated and this means information is being lost and it is not good for the model. A time series model with no serial correlation is called a healthy model.

BIC - Bayesian information criterion, is a criterion for our model selection within the ARIMA and dynamic regression models. BIC indicates complexity of a model, it is a fair trade-off between the forecasting error and the no. of parameters used in the model. This is commonly used for ARIMA related models.

Validation of the Data Models:

Let us take a single restaurant (air_store_id: "air_6d65542aa43b598b") and check the forecast and the validation results for all the models

Exponential Smoothing (ets):

For a restaurant with air_store_id: "air_6d65542aa43b598b", the ets model is validated with the testing set (holdout data).

Accuracy: of the model is calculated by validating with the testing set and the RMSE of the test set is the measure for the accuracy of the model as

shown below:

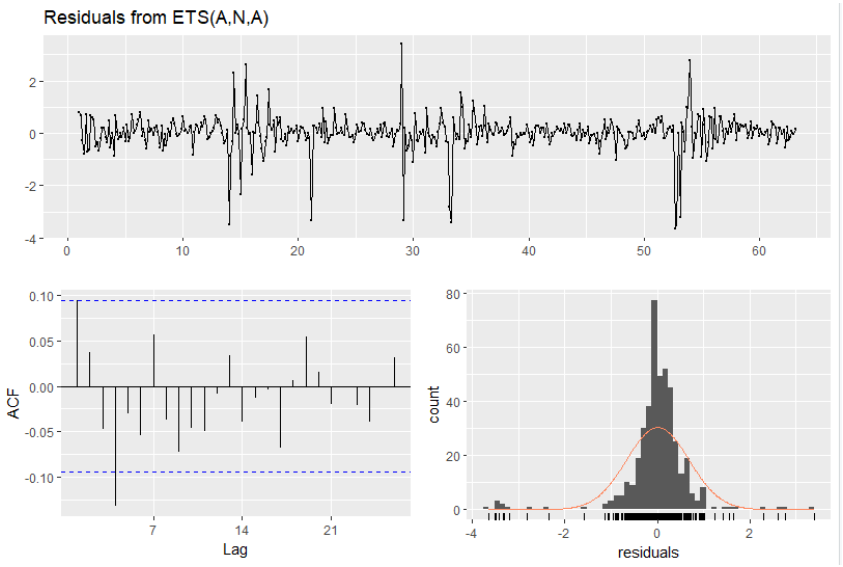
```
> ac_01 <- accuracy(f_cast_01, dtest)
> ac_01
                      ME      RMSE       MAE  MPE MAPE      MASE       ACF1 Theil's U
Training set  0.009250711 0.6737744 0.3827596  NaN  Inf 0.9163079 0.09401105        NA
Test set     -0.064225489 0.9139480 0.3837808 -Inf  Inf 0.9187524 0.07657382         0
```

Predictive Power: of the model is determined by the R^2 of the training and testing set.

```
> test_r2_01 <- cor(f_cast_01$mean, dtest)^2
> train_r2_01
[1] 0.7131994
> test_r2_01
[1] 0.4980285
```

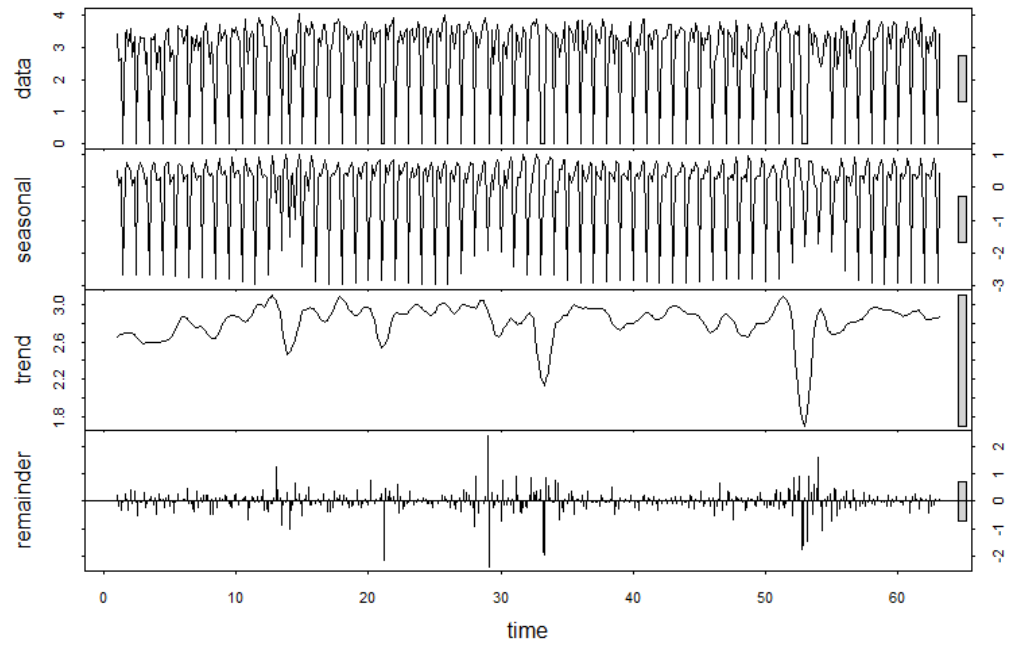
Healthy Model: Checking the residuals and the ACF (Auto Correlation Function) using Ljung- Box Test to detect serial correlation in the model. If there is no serial correlation, then the model is considered healthy. But here, we can see that the error ACF is significant at lags 1 & 4 and this means that there is serial correlation.

Exhibit 5: Check residual plot for LB test



Time Series Decomposition:

Exhibit 6: Decomposed time series



Accuracy: of the model is calculated by validating with the testing set and the RMSE of the test set is the measure for the accuracy of the model

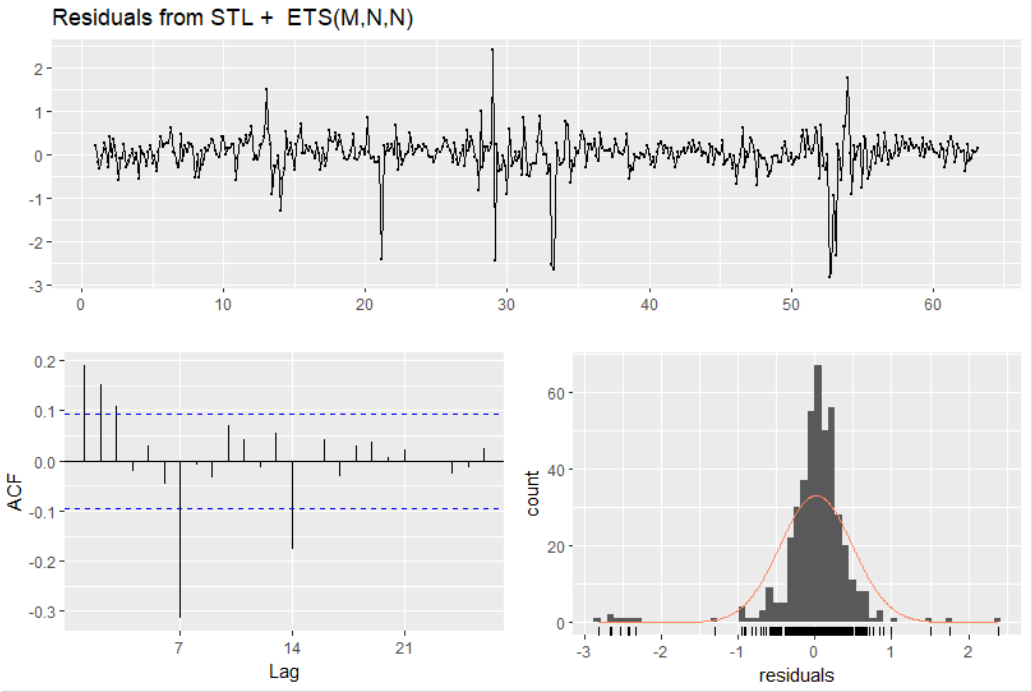
```
> ac_02 <- accuracy(f_cast_02, dtest)
> ac_02
                     ME      RMSE       MAE MPE MAPE      MASE       ACF1 Theil's U
Training set  0.01875587 0.4702363 0.2780076 NaN  Inf 0.6655366 0.18953129        NA
Test set     -0.01965828 0.9179198 0.3948409 NaN  Inf 0.9452300 0.07185199         0
```

Predictive Power: of the model is determined by the R^2 of the training is NA, since the model is not fitted but decomposed.

```
> train_r2_02
[1] "NA for STL"
> test_r2_02
[1] 0.4975661
```

Healthy Model: Checking the residuals and the ACF (Auto Correlation Function) using Ljung- Box Test, we can see that the error ACF is significant at lags 1, 2, 3 & 7 and this means that there is serial correlation.

Exhibit 7: Check residual for LB



Multiple Regression:

Accuracy: of the model is calculated by validating with the testing set and the RMSE of the test set is the measure for the accuracy of the model

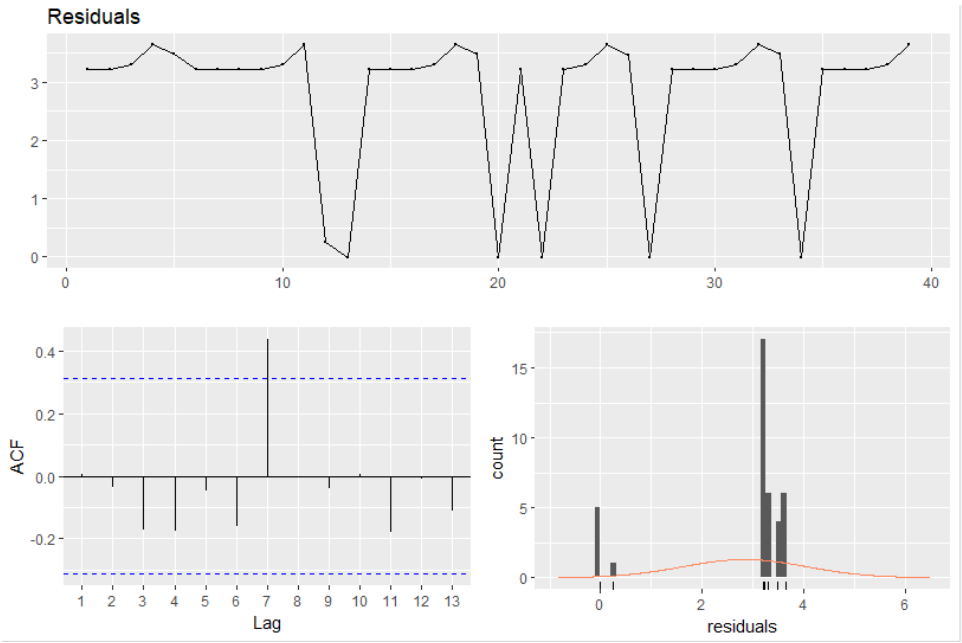
```
> ac_03 <- accuracy(f_cast_03, l_test)
> ac_03
                    ME      RMSE       MAE MPE MAPE
Test set 0.02211628 0.1740445 0.1288005 NaN  Inf
```

Predictive Power: of the model is determined by the R^2 of the training and testing set.

```
> test_r2_03 <- cor(f_cast_03, l_test)^2
> train_r2_03
[1] 0.9491685
> test_r2_03
[1] 0.9807574
```

Healthy Model: Checking the residuals and the ACF (Auto Correlation Function) using Ljung- Box Test, we can see that the error ACF is highly significant at lag 7 and this means that there is serial correlation.

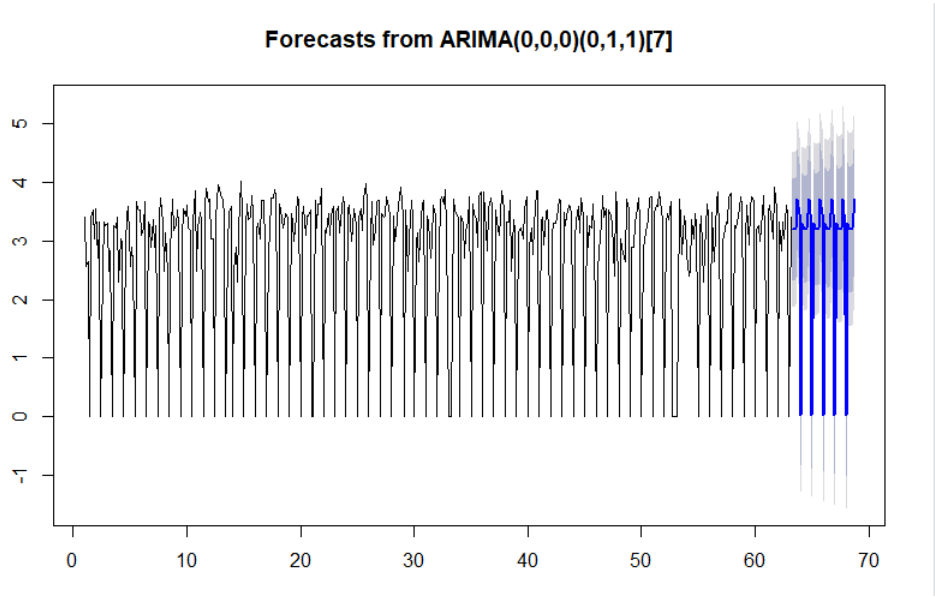
Exhibit 8: Check residuals for LB



ARIMA Model:

Forecast: the plot below shows the forecast for the holdout (in blue) using the ARIMA model of (0,0,0) (0,1,1) for the weekly frequency timeseries

Exhibit 9: Forecast for ARIMA



Accuracy: of the model is calculated by validating with the testing set and the RMSE of the test set is the measure for the accuracy of the model.

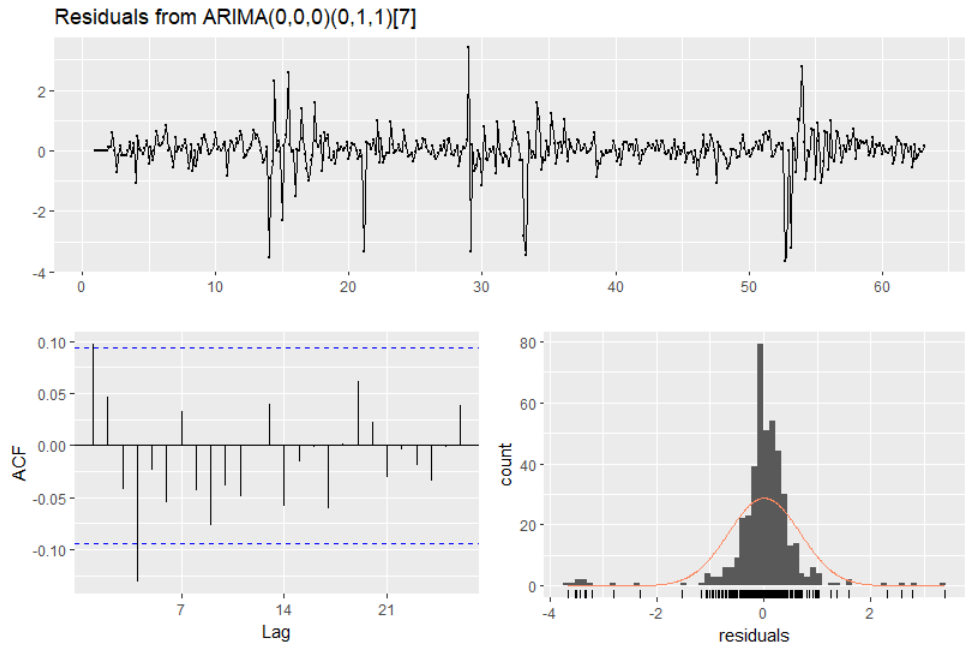
```
> ##checking the accuracy
>   ac_04 <- accuracy(f_cast_04, dtest)
> ac_04
                    ME      RMSE       MAE  MPE MAPE      MASE       ACF1 Theil's U
Training set  0.01022131 0.6654365 0.3643730 -Inf  Inf 0.8722912 0.09693726        NA
Test set     -0.06492310 0.9144991 0.3848122 -Inf  Inf 0.9212218 0.07585681         0
```

Predictive Power: of the model is determined by the R^2 of the training and testing set.

```
> #calculating R2 for train set the model
>   train_r2_04 <- cor(fitted(mod_04), dtrain)^2
> #calculating R2 for test set of the model
>   test_r2_04 <- cor(f_cast_04$mean, dtest)^2
> train_r2_04
[1] 0.7206493
> test_r2_04
[1] 0.4980047
```

Healthy Model: Checking the residuals and the ACF (Auto Correlation Function) using Ljung- Box Test, we can see that the error ACF is significant at lags 1 & 7 and this means that there is serial correlation.

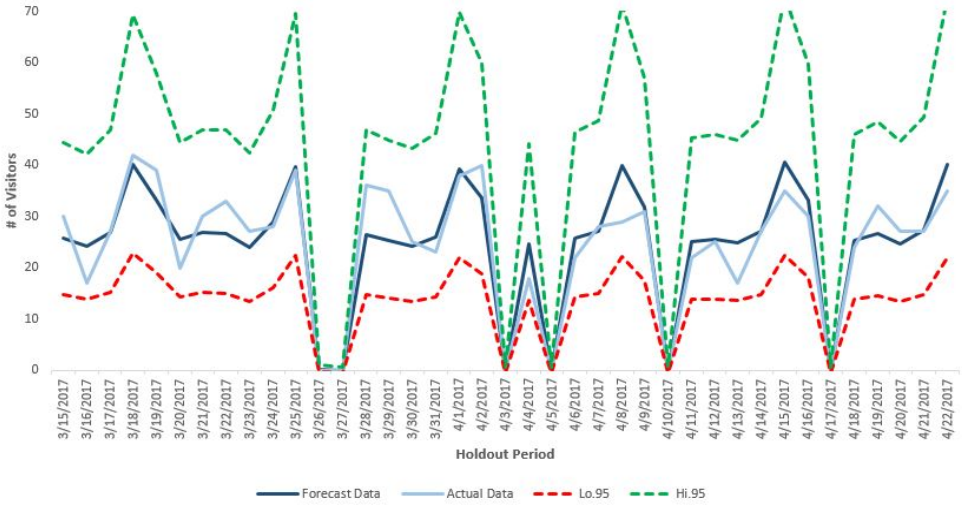
Exhibit 10: Residuals from ARIMA



Dynamic Regression:

Forecast: the plot below shows the forecast for the holdout (in blue) using the ARIMA model of (0,0,0) (0,1,1) for the weekly frequency time series.

Exhibit 11: Point Forecast with confidence interval



Accuracy: of the model is calculated by validating with the testing set and the RMSE of the test set is the measure for the accuracy of the model.

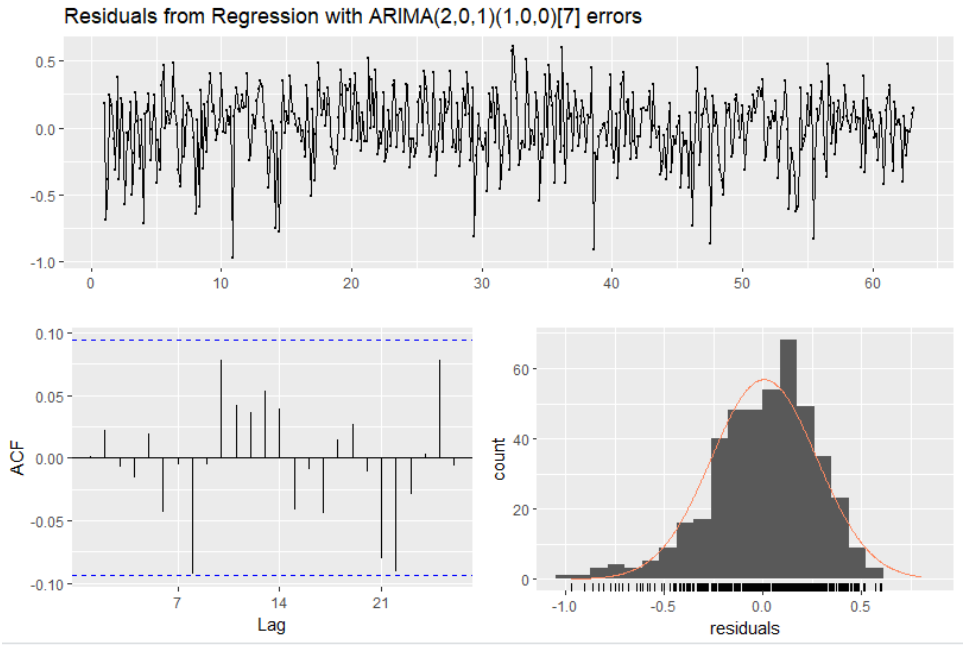
```
> ##checking the accuracy
>   ac_05 <- accuracy(f_cast_05, dtest)
> ac_05
                     ME      RMSE       MAE MPE MAPE      MASE          ACF1 Theil's U
Training set  0.006039652 0.2674039 0.2098107 NaN  Inf 0.5022765 0.0009036901        NA
Test set     -0.010252159 0.1619097 0.1264575 NaN  Inf 0.3027330 0.0055387764         0
```

Predictive Power: of the model is determined by the R^2 of the training and testing set.

```
> #calculating R2 for train set the model
>   train_r2_05 <- cor(fitted(mod_05), dtrain)^2
> #calculating R2 for test set of the model
>   test_r2_05 <- cor(f_cast_05$mean, dtest)^2
> train_r2_05
[1] 0.9544716
> test_r2_05
[1] 0.9832458
```

Healthy Model: Checking the residuals and the ACF (Auto Correlation Function) using Ljung- Box Test, we can see that the error ACF is not significant at any lags and this means that there is no serial correlation. The model is healthy.

Exhibit 12: Residuals from regression with ARIMA



Compared to all the models, regression with ARIMA (2,0,1) (1,0,0)[7] had the lowest RMSLE and it implies that the R^2 of the model should be good (R^2 = 0.98). And so the model selection algorithm selected this model for the restaurant with id: "air_6d65542aa43b598b"

air_store_id	Model_No.	Selected_Model	Training_R2	Testing_R2	RMSLE
air_6d65542aa43b598b	model_5	Regression with ARIMA(2,0,1)(1,0,0)[7] errors	0.954471575	0.983245827	0.161909734

On manual analysis of the generated LB test results, we can also see that the model is healthy since there is no serial correlation in the selected model and this is an optional criterion.

Output of the Data Modeling and Selection:

The same validation and model selection process is followed for all the 829 restaurants and a suitable forecasting model is selected for forecasting the no. of visitors for the next 39 days in our Kaggle problem's objective (till 31st May 2017). For this automated selection process, Healthy model criteria is not strictly followed, since it required manual analysis of the Ljung-Box test (LB) as elaborated in the previous section.

The output of the automated data selection algorithm we built in R shows the 829 restaurants with best models selected for them and printed as shown below.

air_store_id	Model_No.	Selected_Model	Training_R2	Testing_R2	RMSLE
air_d63cfa6d6ab78446	model_5	Regression with ARIMA(0,0,0)(1,0,0)[7] errors	0.965079897	0.982447527	0.121495148
air_2703dcb33192b181	model_3	l_train ~ closed_days + mon_thurs + friday + saturday	0.983829429	0.994466301	0.126791963
air_105a7954e32dba9b	model_3	l_train ~ closed_days + mon_thurs + friday + saturday	0.994805057	0.995785113	0.14206339
air_694571ea13fb9e0e	model_2	STL + ETS(A,N,N)	NA for STL	0.991422787	0.150461993
air_6d65542aa43b598b	model_5	Regression with ARIMA(2,0,1)(1,0,0)[7] errors	0.954471575	0.983245827	0.161909734
air_55390f784018349a	model_5	Regression with ARIMA(0,1,2) errors	0.984553495	0.985346212	0.162163996

The output from our model selection algorithm for all the 829 restaurants has an average test set RMSLE of 0.449. And this beats the Kaggle's price RMSLE of 0.51.

Q1. Forecast on the number of visitors in a day for a given restaurant

Using the forecasting model selected for the restaurants, we can forecast the no. of visitors (taking anti-log of log_visitors) for each restaurant with more accuracy (least errors). With this, we will be able to anticipate the total no. of visitors on a given day for a restaurant and better manage our business.

Business interpretation for the time Series Model:

Restaurant planning is a complex daily task faced by restaurant managers. The managers juggle in food freshness commitment, staff scheduling, guest traffic, promotions, and competitors. With this data-driven forecasting model managers simplifies planning and forecasting while improving productivity and reducing food waste for the restaurant each day. The forecasting results from the model give advantages in following ways:

Greater operational efficiency: The forecast data ensures kitchen filled with required ingredients and food products to meet the daily demand. This way managers can handle the excess demands on promotions or special events based on past knowledge. The accurate forecast gives benefits of inventory management, financial planning, and marketing of the restaurant.

Enhanced guest service: Customer service is the key to success in the restaurant industry. The accurate forecast gives the estimation of stock in

the kitchen and fewer lost sales which lead to higher customer satisfaction. The improved planning helps in focusing on creating value for customers.

Higher productivity: Managers can schedule staff more accurately with visitors forecasting and increase employee engagement. Also, the manager can hire part-time staff based on the promotion, events or special days.

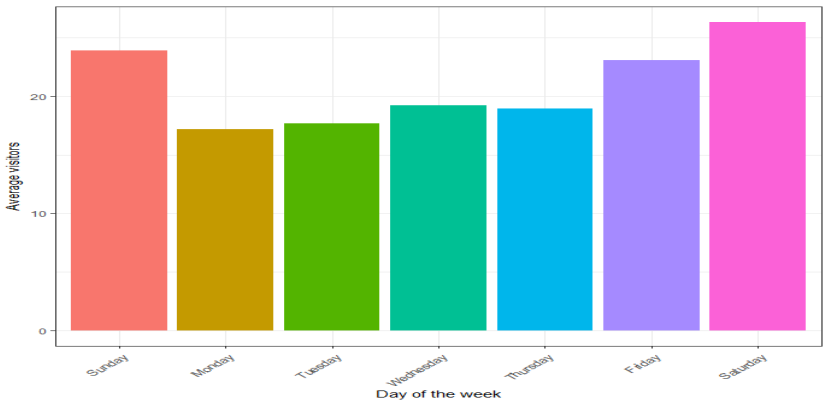
Reduce food waste: By tracking inventory and sales estimation based on visitors forecasting managers can increase production accuracy, control the food costs and improve the profit margins

Q2. Predict the busiest day, week and month for a given restaurant.

From the model selected for forecasting, we can forecast the no. of visitors that might come for a given day for each restaurant.

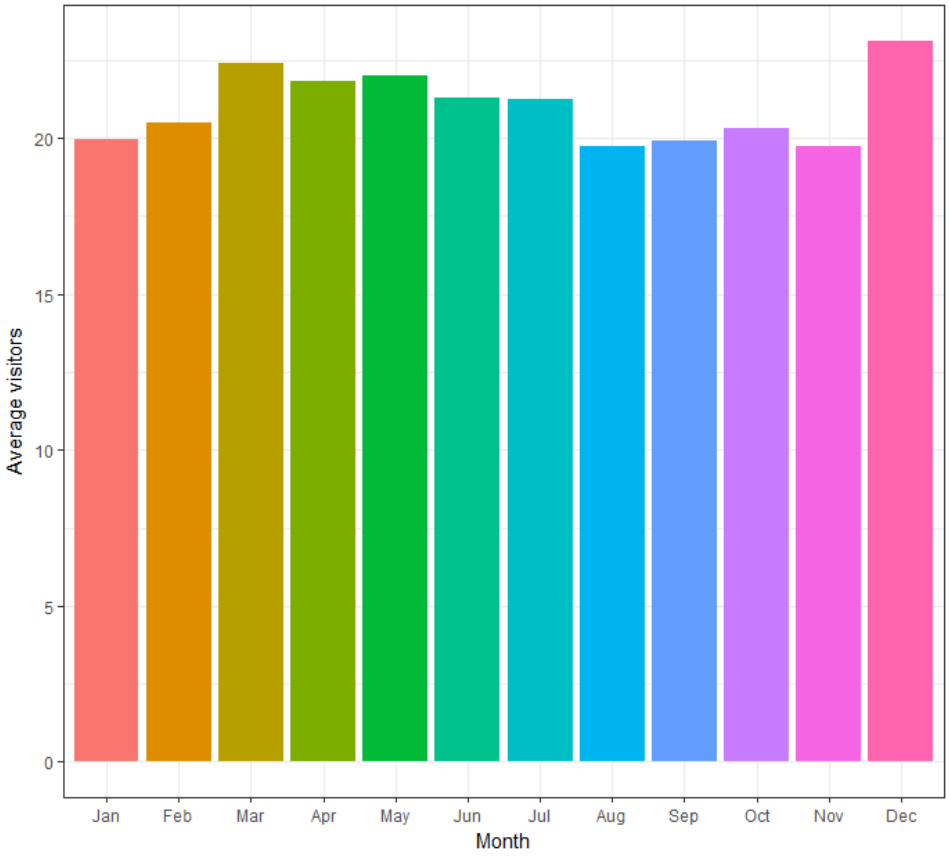
The exhibit 13 shows the average no. of visitors for all the restaurants. From this, we can see that the busiest days of the week are Saturday, Sunday and Friday (in that order). The average number of visitors are over 25 on Saturday.

Exhibit 13: Busiest Day of the week



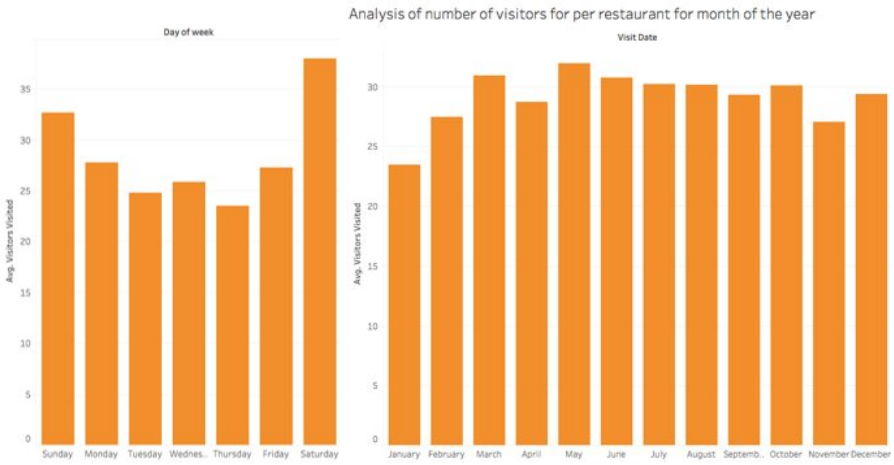
From the Exhibit 14, we can see that the busiest month of the year is December followed by March. This increase in the average number of visitors must correspond to the holiday season. The number of reservation spike in December because of Christmas and new year. The spike in the number of visitors in March is due to international tourists, who visit Japan during spring because of their widely known cherry blossom season.

Exhibit 14: Busiest Day of the month



The Exhibit 15 below shows that the busiest day of the week and the month of the week for a restaurant "air_6d65542aa43b598b". This follows the same pattern; the busiest day of the week is the weekend and busiest time of the month is December and spring season.

Exhibit 15: Day of the week and month of year for one restaurant



Q4. Predict how many reservations get converted into actual visits

Using the model built, the reservations for a given restaurant helps predict the number of visits. The model

To understand the relationship between reservation and visits, individual analysis of the data was done first.

Analysis of Air Visits (air_visit_data):

The Air Visit data contains information on the number of visits to each restaurant. The information is captured daily starting from January 2016 to April 2014. As shown in the snippet below

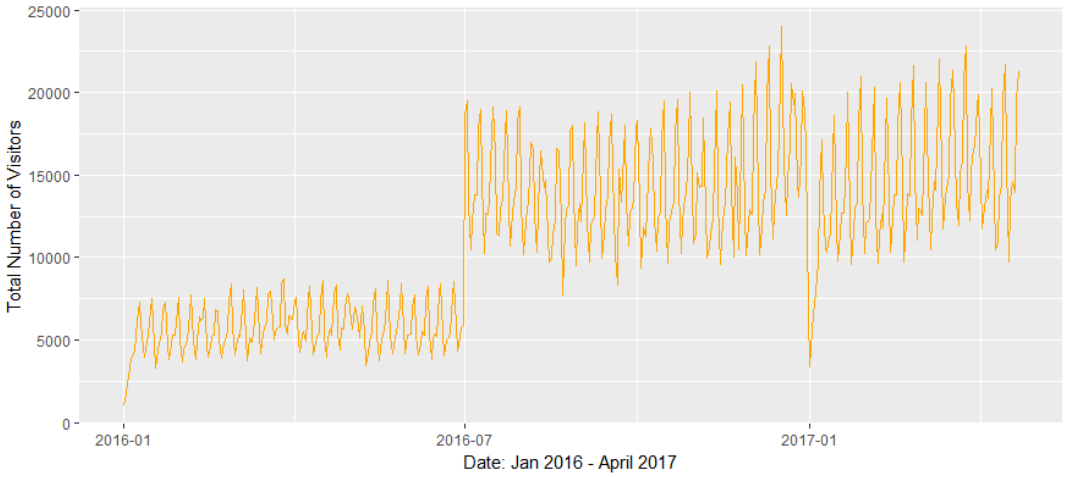
- There is a total of 252,108 records

- The Air Visit data contains 3 variables: - "air_store_id", "visit_date", "visitors"
- The visit data, that is, number of visitors are capture for 829 unique restaurants

```
> summary(air_visits)
               air_store_id      visit_date            visitors
 air_5c817ef28f236bdf:   477   Min.   :2016-01-01   Min.   :  1.00
 air_36bcf77d3382d36e:   476   1st Qu.:2016-07-23   1st Qu.:  9.00
 air_a083834e7ffe187e:   476   Median :2016-10-23   Median : 17.00
 air_d97dabf7aae60da5:   476   Mean   :2016-10-12   Mean   : 20.97
 air_232dcee6f7c51d37:   475   3rd Qu.:2017-01-24   3rd Qu.: 29.00
 air_60a7057184ec7ec7:   475   Max.   :2017-04-22   Max.   :877.00
 (Other)             :249253
> glimpse(air_visits)
Observations: 252,108
Variables: 3
$ air_store_id <fct> air_ba937bf13d40fb24, air_ba937bf13d40fb24, air_ba937bf13d40fb24, air_ba
$ visit_date   <date> 2016-01-13, 2016-01-14, 2016-01-15, 2016-01-16, 2016-01-18, 2016-01-19,
$ visitors     <int> 25, 32, 29, 22, 6, 9, 31, 21, 18, 26, 21, 11, 24, 21, 26, 6, 18, 12, 45,
>
> air_visits %>% distinct(air_store_id) %>% nrow()
[1] 829
```

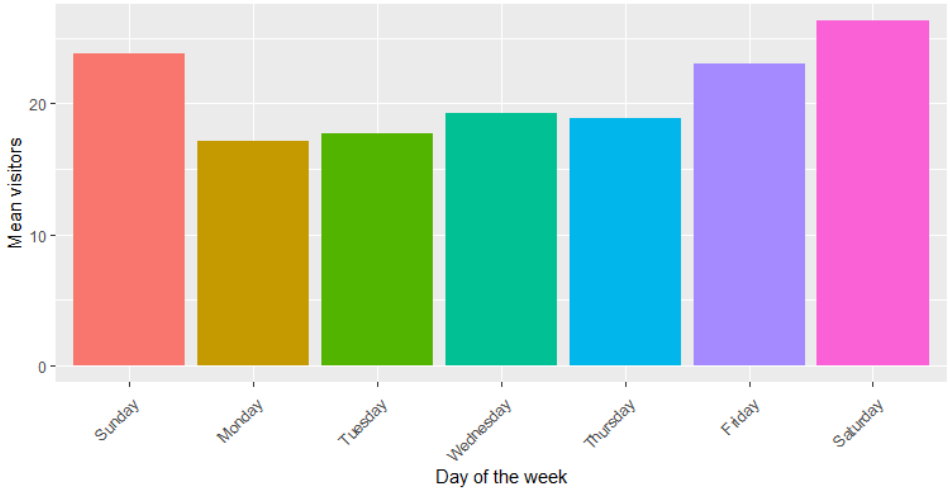
Plotting the total number of visitors over the time-period shows that there are spikes in the total number of visitors. This must be because new restaurants are getting added to the database. We also see periodic patterns that corresponds to weekly cycle.

Exhibit 16: Time series for Air Visits



By aggregating the data to day of the week, we see that median number of visitors is over 20. As expected, the busiest days are Friday, Saturday and Sunday. Monday and Tuesday have the lowest number of visitors.

Exhibit 17: Mean visitors for day of the week



Since the reservation are made through both the AIR system and the HPG system, separate analysis was made on both the systems.

Analysis of Air Reservation (air_reserve):

The air reservation data contains information on when the reservation was made with date and time stamp, the date and timestamp on when the visit for that reservation will occur and the number for visitor for that reservation for a restaurant. This data is collected daily for each restaurant. As shown in the snippet below

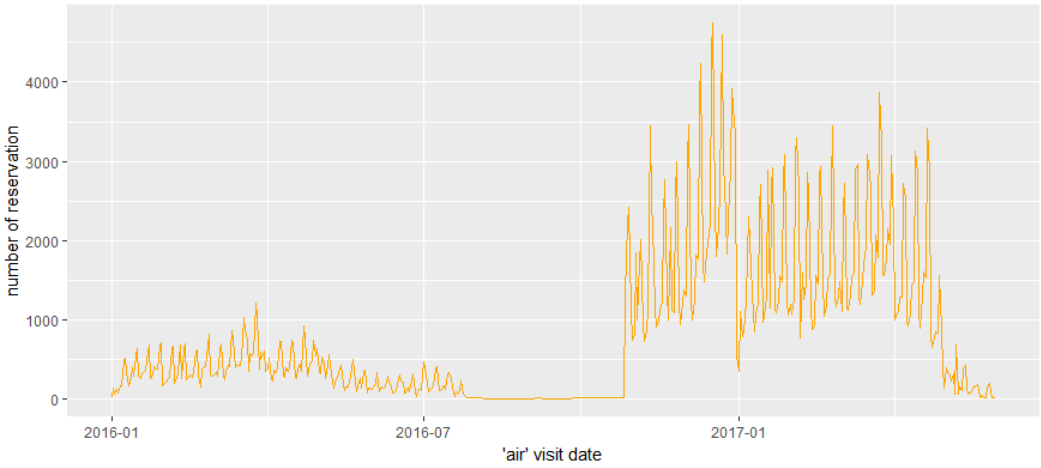
- There is a total of 92,378 records
- The Air reservation file contains 4 variables: - "air_store_id", "visit_datetime","reserve_datetime" and "reserve_visitors"
- The reserve data, that is, number of reserves are capture for 314 unique restaurants

```
> summary(air_reserve)
         air_store_id   visit_datetime                reserve_datetime              reserve_visitors
 air_8093d0b565e9dbdf: 2263   Min.   :2016-01-01 19:00:00   Min.   :2016-01-01 01:00:00   Min.   :  1.000
 air_e55abd740f93ecc4: 1903   1st Qu.:2016-11-15 19:00:00   1st Qu.:2016-11-07 17:00:00   1st Qu.:  2.000
 air_0a74a5408a0b8642: 1831   Median :2017-01-05 18:00:00   Median :2016-12-27 22:00:00   Median :  3.000
 air_cf5ab75a0afb8af9: 1758   Mean   :2016-12-05 08:18:58   Mean   :2016-11-27 01:13:07   Mean   :  4.482
 air_6d65542aa43b598b: 1436   3rd Qu.:2017-03-03 19:00:00   3rd Qu.:2017-02-26 18:00:00   3rd Qu.:  5.000
 air_de692863bb2dd758: 1355   Max.   :2017-05-31 21:00:00   Max.   :2017-04-22 23:00:00   Max.   :100.000
 (Other)             :81832
> glimpse(air_reserve)
Observations: 92,378
Variables: 4
$ air_store_id     <fct> air_877f79706adbfb06, air_db4b38ebe7a7ceff, air_db4b38ebe7a7ceff, air_877f79706adbfb
$ visit_datetime   <dttm> 2016-01-01 19:00:00, 2016-01-01 19:00:00, 2016-01-01 19:00:00, 2016-01-01 20:00:00,
$ reserve_datetime <dttm> 2016-01-01 16:00:00, 2016-01-01 19:00:00, 2016-01-01 19:00:00, 2016-01-01 16:00:00,
$ reserve_visitors <int> 1, 3, 6, 2, 5, 2, 4, 2, 2, 2, 3, 3, 2, 6, 7, 41, 13, 2, 3, 2, 2, 3, 2, 2, 6, 2, 2, 4
>
> air_reserve %>% distinct(air_store_id) %>% nrow()
[1] 314
```

From the below graph, we can see that there are fewer reservation in 2016. There is a period when there are no reservations in 2016. In 2017, the reservation increases. The artificial decline after the first quarter is most likely related to these reservations being at the end of the training time

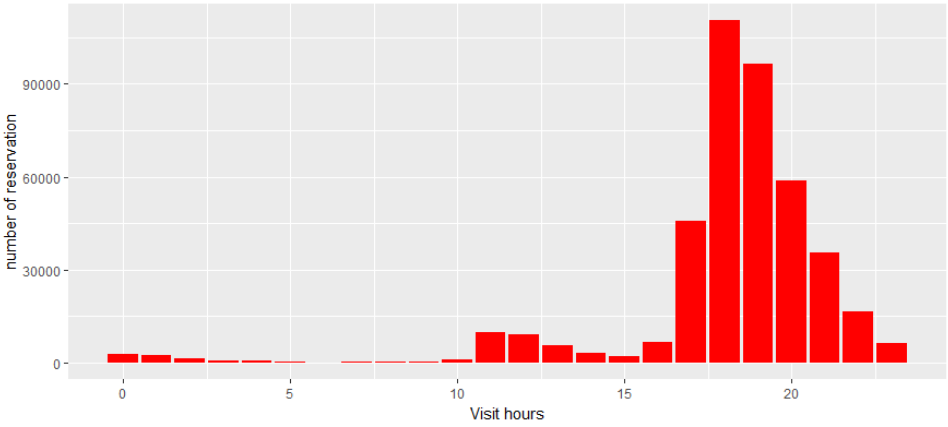
frame, which means that long-term reservations would not be part of this data set.

Exhibit 18: Time series for Air Reservation



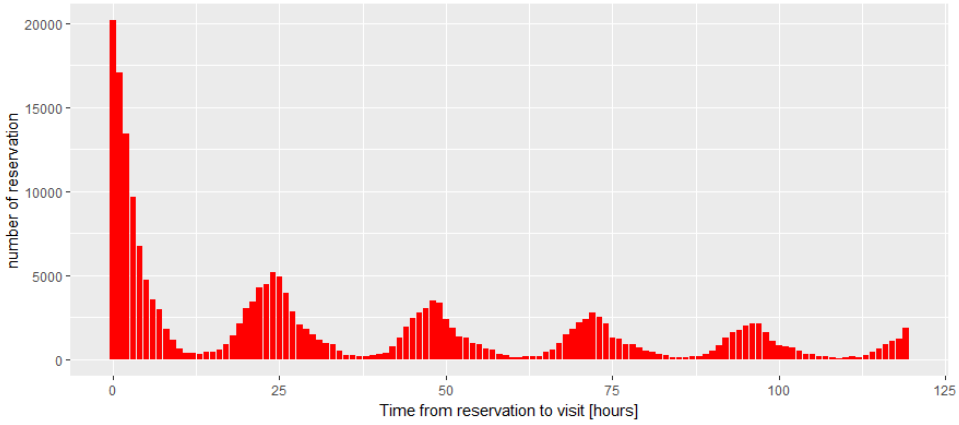
By plotting visit hours against the number of visitors, we can see that most of the reservation are made at evening close to dinner time.

Exhibit 19: Air reservation time of visit



Also, by looking at the time of reservation and number of visitors, we can deduce that most of the reservation close to the visit time. There fewer reservations made in the future. The number of visitors for future reservation are also less when compared to the reservation made a few hours before visit.

Exhibit 20: Air reservation visits against time of reservation



Analysis of HPG Reservation (hpg_reserve):

Like Air reserve data the HPG reserve data holds information on when the reservation was made on the HPG site with date and time stamp, the date and timestamp on when the visit for that reservation will occur and the number for visitor for that reservation for a restaurant. This data is collected daily for each restaurant. As shown in the snippet below

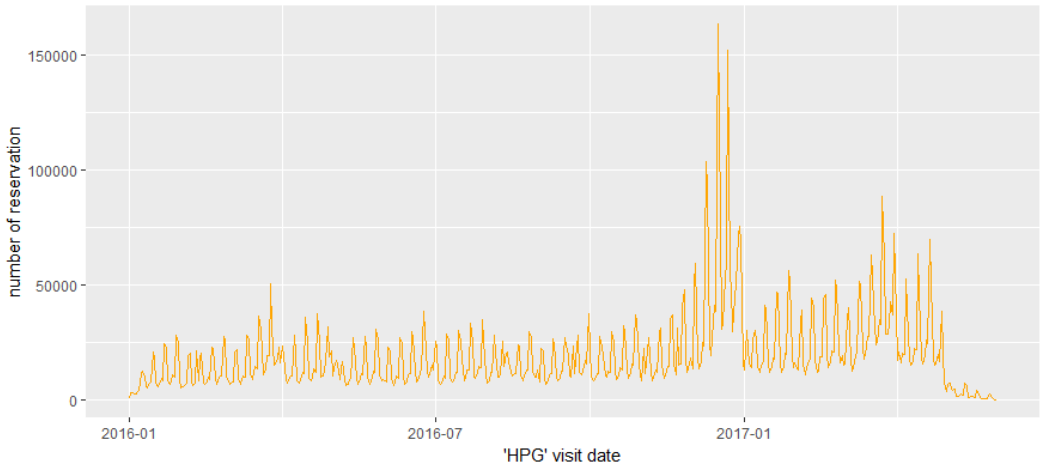
- There is a total of 2, 000,320 records

- The Air reservation file contains 4 variables: - "air_store_id", "visit_datetime","reserve_datetime" and "reserve_visitors"
- The reserve data, that is, number of reserves are capture for 13325 unique restaurant

```
> summary(hpg_reserve)
               hpg_store_id     visit_datetime                reserve_datetime              reserve_visitors
 hpg_2afd5b187409eeb4:   1155   Min.   :2016-01-01 11:00:00   Min.   :2016-01-01 00:00:00   Min.   :  1.000
 hpg_011e799ba201ad2e:    822   1st Qu.:2016-06-26 19:00:00   1st Qu.:2016-06-21 12:00:00   1st Qu.:  2.000
 hpg_9b20c78a9b8179d9:    778   Median :2016-11-19 20:00:00   Median :2016-11-10 20:00:00   Median :  3.000
 hpg_527c60506b80ac72:    740   Mean   :2016-10-15 06:55:20   Mean   :2016-10-07 19:57:59   Mean   :  5.074
 hpg_3f9e56ac6f9435c7:    729   3rd Qu.:2017-02-03 19:00:00   3rd Qu.:2017-01-27 13:00:00   3rd Qu.:  6.000
 hpg_4ca09101fa3a220c:    712   Max.   :2017-05-31 23:00:00   Max.   :2017-04-22 23:00:00   Max.   :100.000
 (Other)             :1995384
> glimpse(hpg_reserve)
Observations: 2,000,320
Variables: 4
$ hpg_store_id     <fct> hpg_c63f6f42e088e50f, hpg_dac72789163a3f47, hpg_c8e24dcf51ca1eb5, hpg_24bb207e5fd49d4a, hp.
$ visit_datetime   <dttm> 2016-01-01 11:00:00, 2016-01-01 13:00:00, 2016-01-01 16:00:00, 2016-01-01 17:00:00, 2016-.
$ reserve_datetime <dttm> 2016-01-01 09:00:00, 2016-01-01 06:00:00, 2016-01-01 14:00:00, 2016-01-01 11:00:00, 2016-.
$ reserve_visitors <int> 1, 3, 2, 5, 13, 2, 2, 2, 2, 6, 2, 2, 2, 2, 5, 4, 2, 4, 5, 2, 5, 4, 5, 3, 2, 4, 4, 2, 2, 7,.
> hpg_reserve %>% distinct(hpg_store_id) %>% nrow()
[1] 13325
```

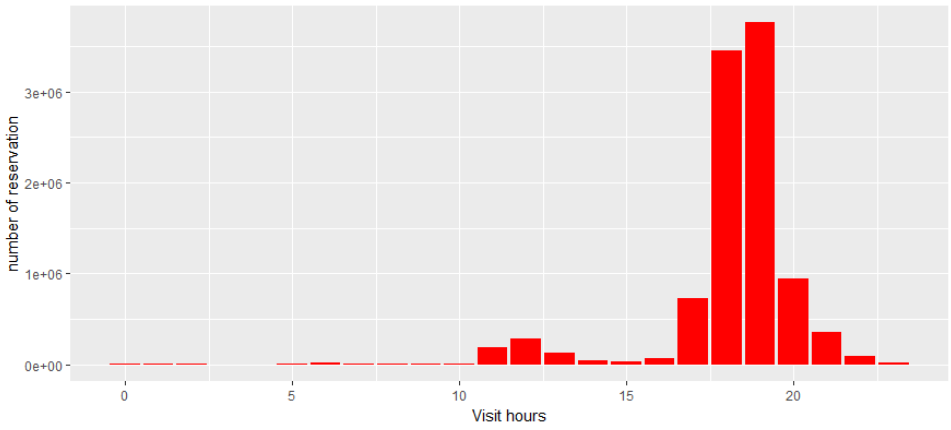
We can see that there is sudden spike in the number of reservation in December 2016.

Exhibit 21: HPG reservation time series



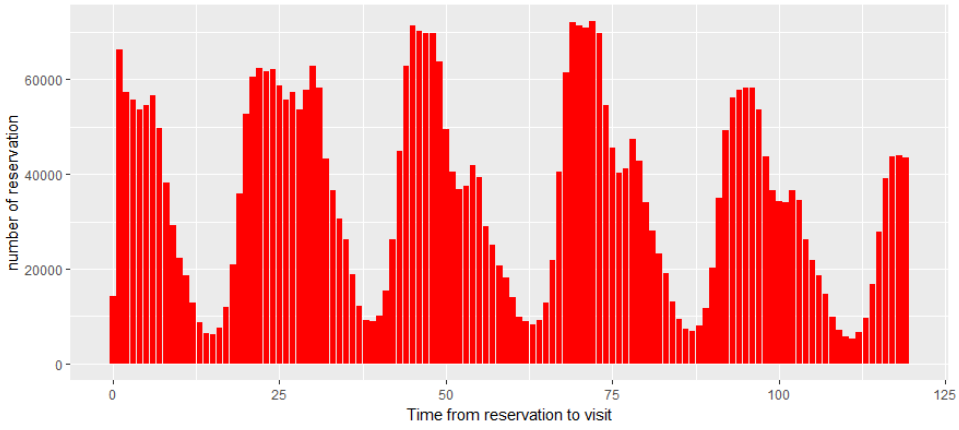
Like Air reservations, most of the reservations are made in the evening close to dinner time.

Exhibit 22: HPG reservation time of visit



But contrast to Air reserves, we see that there is no huge change in number of reservations who make reservations early or in the future. We see that there is dip in the number of reservations for reservations made almost four days or more in the future.

Exhibit 23: HPG reservation visits against time of reservation



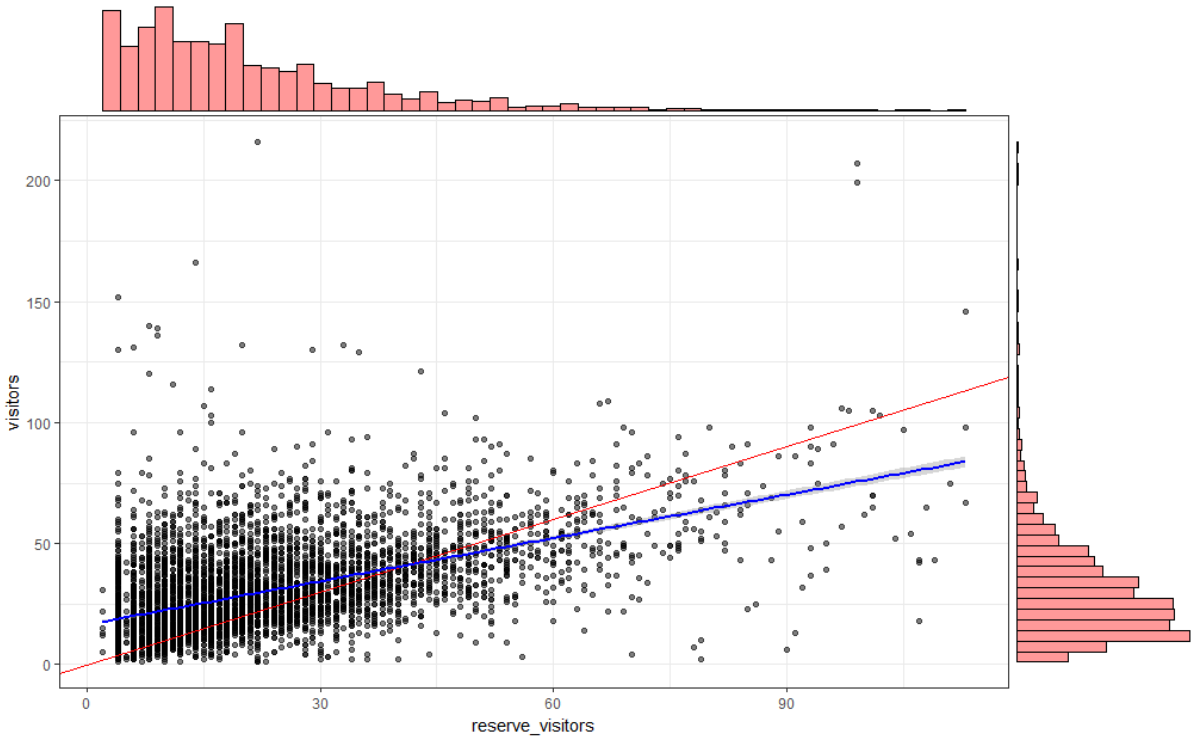
Analysis of Reserve Vs Visits:

To compare the reservations with visits, first we combined the reservation made through the Air system and HPG system using the store id relation file. This gives the total number of reservation made to a restaurant.

By plotting a scatter plot with marginal histogram of total number of reservations and the number of visitors, the following findings were made.

- There are more number of visitors than the number of reservations. This is because there was more walk-in customer
- Since there are a lot of scatter point below the line, there were customers who made reservation but did not go.
- The liner fit line shows that the number of reservations does not influence the number of visitors. This because when the reservations are canceled the tables are given to walk-in customers. So, this implies that even if reserves customer did not turn up, we have walk-in customers who fill up the seats.

Exhibit 24: Reservation vs Visits

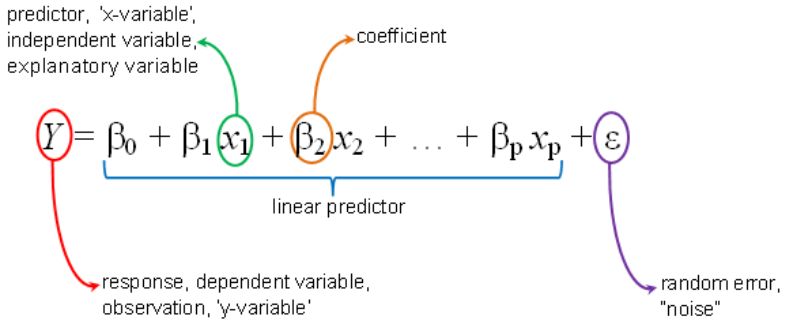


Data Modeling with Python:

Regression:

Regression equation is the mathematical formula is applied to the explanatory variables to best predict the dependent variable. Regression analysis is often used for prediction of a variable and thus answers the why question. In this particular case prediction of visitors to restaurants from the explanatory variables is the question. A typical expression

representing the elements of an Ordinary Least Squares Regression (OLS) is illustrated below in figure 25.



Regression using Python:

Ordinary Least Squares Regression model from Sci kit machine learning library in Python is utilized. A code snippet is shown below for reference.

```
from sklearn import linear_model
model = linear_model.LinearRegression()
```

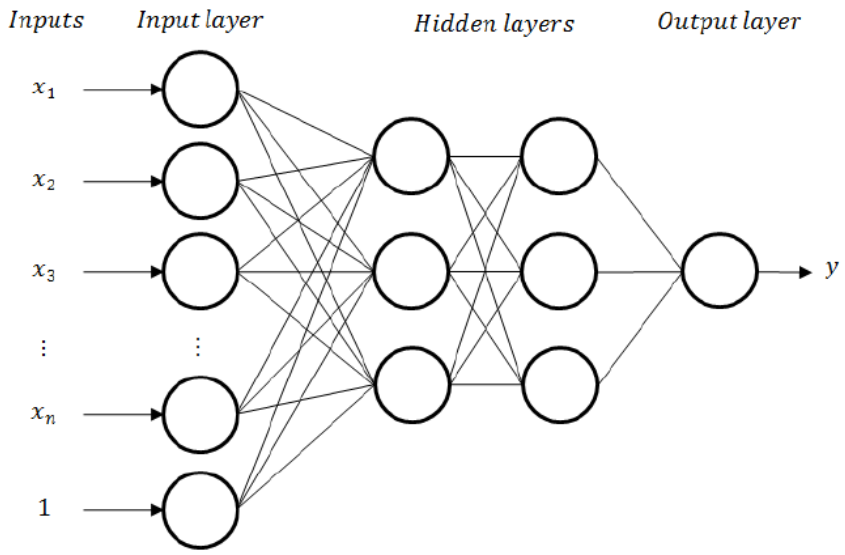
Neural Networks:

The idea behind neural networks is to simulate the neurons similar to human brain, thus when simulated in a computer it's a black box. Input values from the left side are fed to the model and in turn the model assigns weights according to the connections. The output from one layer is transferred to the next layer. This procedure of feeding neurons can be

done in both feed forward and feedback ward direction known as forward or backward propagation.

This procedure happens in many combinations/iterations till the machine learning algorithm arrives at the best possible values. Implementation of predictive MLP neural network algorithm using Python scripting language is explained in next section.

Exhibit 25: Neural network model



Neural Networks using Python:

Multi-layer Perceptron (MLP) supervised learning algorithm from Sci kit machine learning library in Python is utilized for Neural Networks. Before applying the Neural Networks, data was normalized using built in standard scalar method for better conversion of iterations.

Multi-layer Perceptron (MLP) for regression was utilized to predict the number of visitors for a new (unobserved) input from test set. Parameters required to create an instance for the model is shown in the snippet below.

A hidden three-layer neuron network consisting 200, 150 and 100 neurons in each layer were utilized. relu activation function with adam solver was employed.

```
from sklearn.neural_network import MLPRegressor
model = MLPRegressor((200,150,100), activation='relu', solver='adam', batch_size=28, verbose=True, max_iter=1000)
```

Note: Results from respective Regression and Neural Network models are discussed in the Question 3 and Question 5 research questions.

Q3. What type of restaurant (cuisine) to open for a successful business?

As per the data set compiled restaurants in Japan mainly serve 14 types of cuisines. To gather insights on performance of the cuisines a bar chart was plotted of average number of visitors interested in a cuisine as shown in the figure number # below. Some of the noteworthy top performing cuisines are Asian, Karaoke/Party, Creative Cuisine etc. Further, a comparison of Total visitors and visitors walk in as per restaurant cuisines was visualized. Maximum number of visitors prefer walking in to restaurants to eat the cuisines they like instead of reservation.

Exhibit 26: Bar chart of average visitor vs 14 types of cuisines

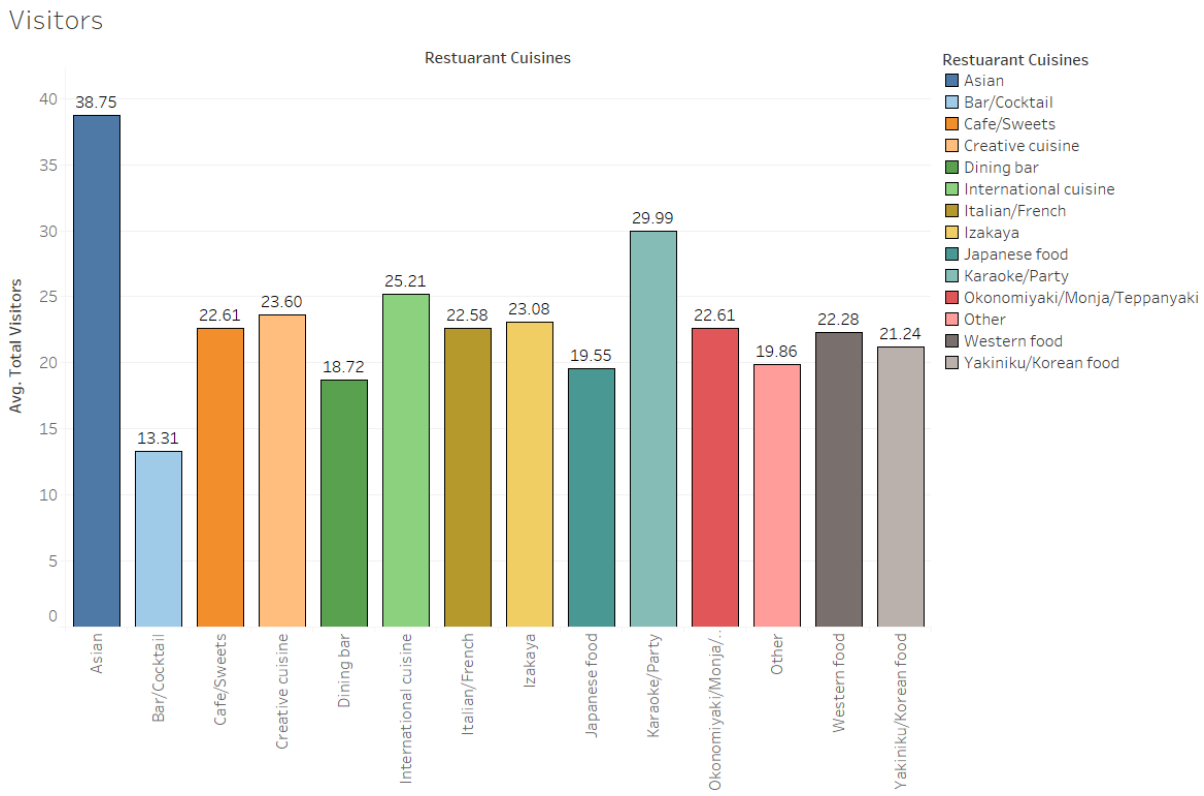
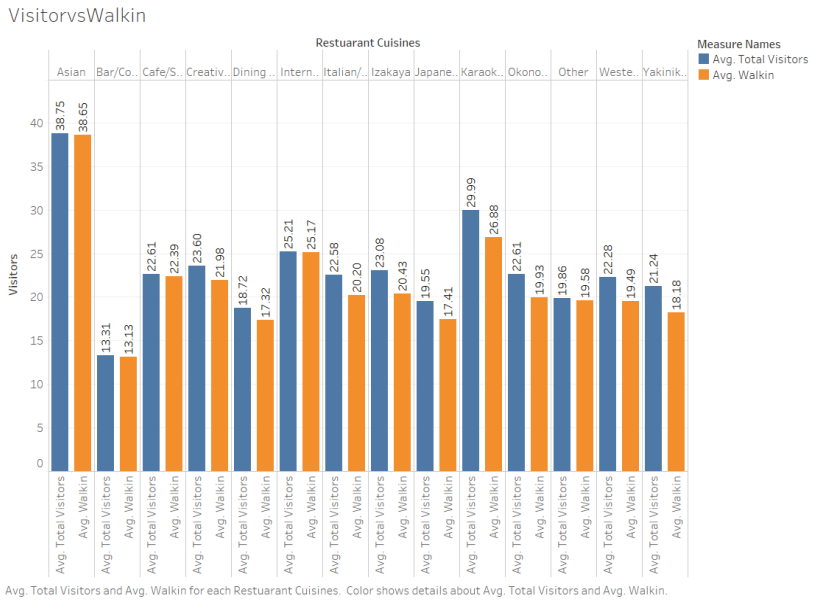


Exhibit 27: Bar chart of Average Total Visitors vs Total Walk in



Results and Discussion

Prediction using Regression and Neural Network models were implemented for few of the selected cuisines. Accuracy of the models were checked by Adjusted R square and RMSE models as tabulated below.

Regression		
Genre	**R Squared**	**RMSE**
Asian	0.813907	0.458667
Creative_Cusine	0.390336	0.843469
Japanese	0.259164	0.632235

R squared and RMSE of Visitor prediction for Regression Model

Neural Networks		
Genre	**R Squared**	**RMSE**
Asian	0.692277	0.58981
Dining	0.443809	0.438031
Japanese	0.646687	0.436734

R squared and RMSE of Visitor prediction for Neural Network

Model

Plots of Actual visitors (Y_Test) vs Predicted visitors (Y_Pred) were plotted for Regression and Neural Network models. Apart from point prediction of visitors a 95% confidence interval of prediction was plotted with a range

of lower limit to upper limit. By plotting 95% confidence interval, and we can say that there is only a 5% chance that the range of visitor predicted values excludes the mean.

Exhibit 28: Actual visitors (Y_Test) vs Predicted visitors (Y_Pred) Asian Cuisine

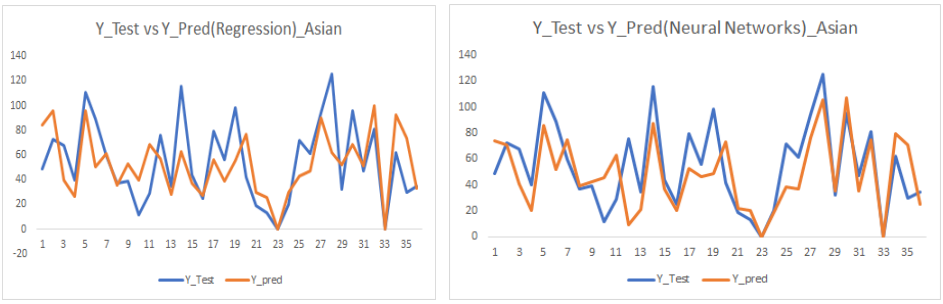


Exhibit 29: 95% confidence interval of Prediction Asian Cuisine

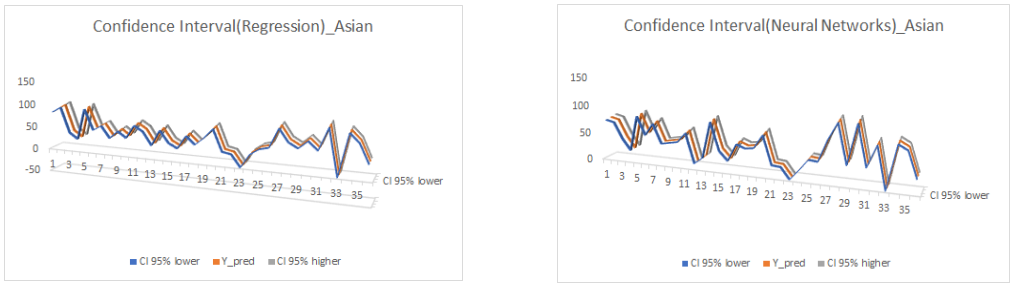


Exhibit 30: Actual visitors (Y_Test) vs Predicted visitors (Y_Pred) Creative Cuisine

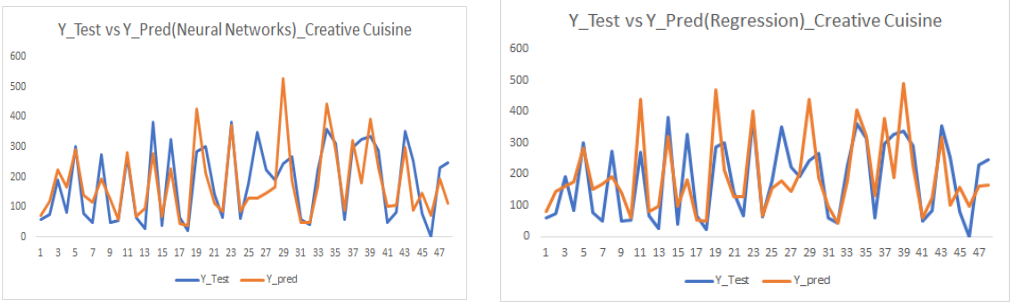
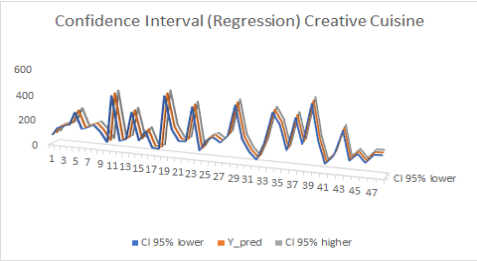
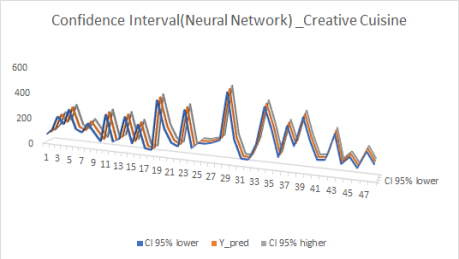


Exhibit 31: 95% confidence interval of Prediction Creative Cuisine





Business Interpretation:

- Adjusted R Square and RMSE values help the managers to judge the quality of prediction to make decisions.
- Business owners will be able to predict which cuisines are selling more based on visitor affinity thus planning the inventory required earlier. This is especially useful during holidays and weekends.
- Confidence interval of the predicted visitors cater for the best, middle and worst-case business scenarios and provides options for the business owners.

Q5. Predicting and visualizing which restaurants has the most number of visitors based on the location data given.

The data set consists of location data for approximately 829 Japanese Restaurants. Top ten restaurants with most number of visitors were selected and plotted on the geographical map using Tableau as shown in

the figure below. Similarly, the areas with most number of visitors was visualized via a tree line graph as shown in the fig below.

Exhibit 32: Top 10 Restaurants based on location

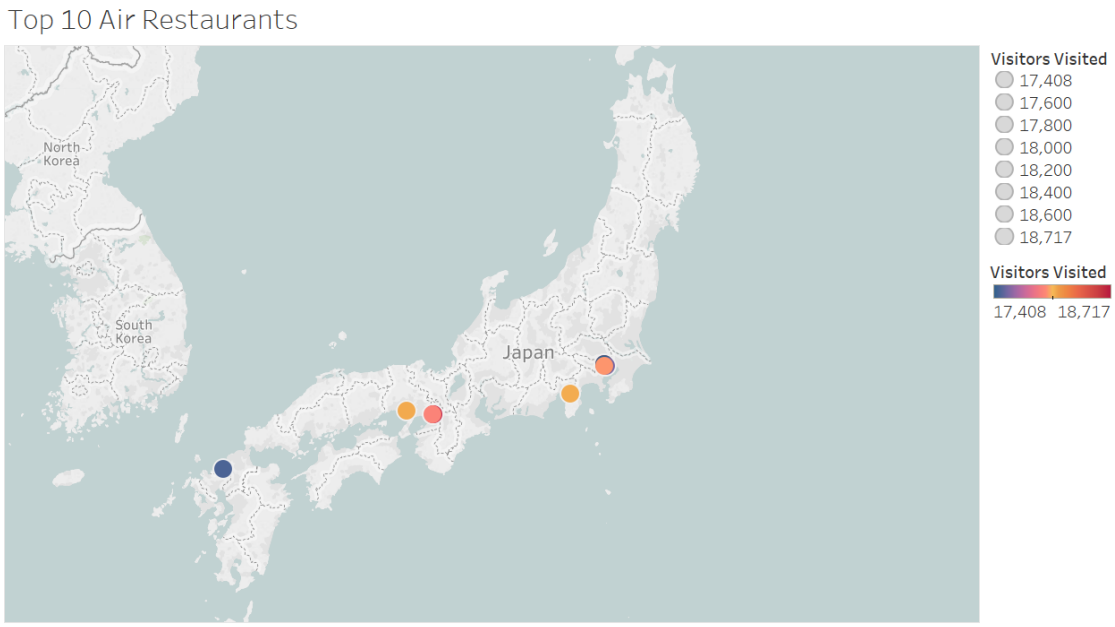
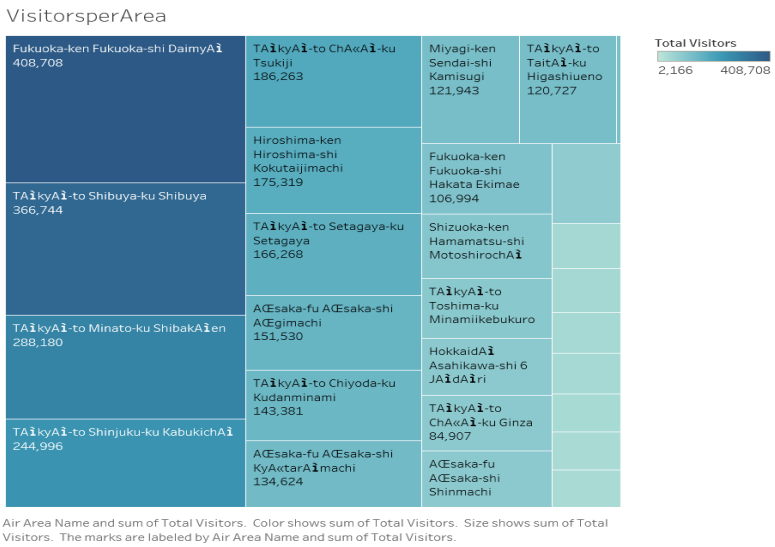


Exhibit 33: Heatmap of visitor per Area



Results and Discussion

From the restaurants with most number of restaurants a few were selected for building the prediction models. R Squared and RMSE values for the restaurants are tabulated as below.

Regression		
Restaurant	**RMSE**	**R Squared**
air_f26f36ec4dc5adb0	0.2784	0.9122
air_e55abd740f93ecc4	0.3906	0.9124
air_99157b6163835eec	0.3858	0.9276

R squared and RMSE of Visitor prediction for Regression Model

Neural Networks		
Restaurant	**RMSE**	**R Squared**
air_f26f36ec4dc5adb0	0.4033	0.8158
air_e55abd740f93ecc4	0.3098	0.9449
air_99157b6163835eec	0.4049	0.9202

R squared and RMSE of Visitor prediction for Neural Network Model

Exhibit 34: Actual visitors (Y_Test) vs Predicted visitors (Y_Pred)

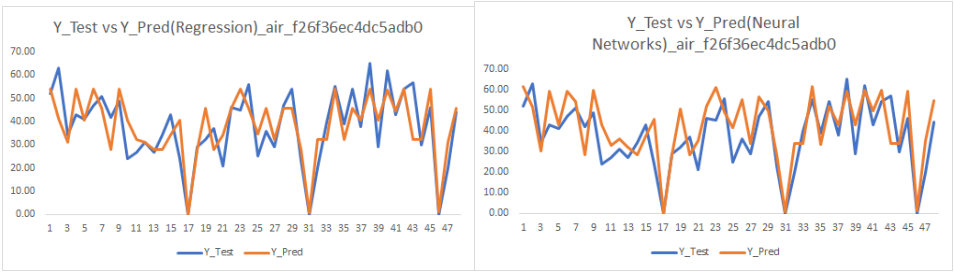


Exhibit 35: 95% confidence interval of Prediction

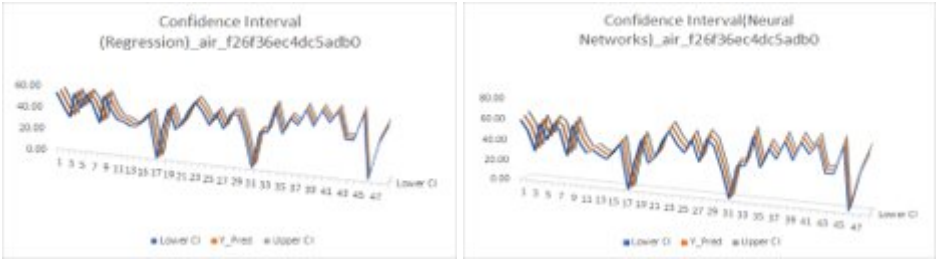


Exhibit 36: Actual visitors (Y_Test) vs Predicted visitors (Y_Pred)

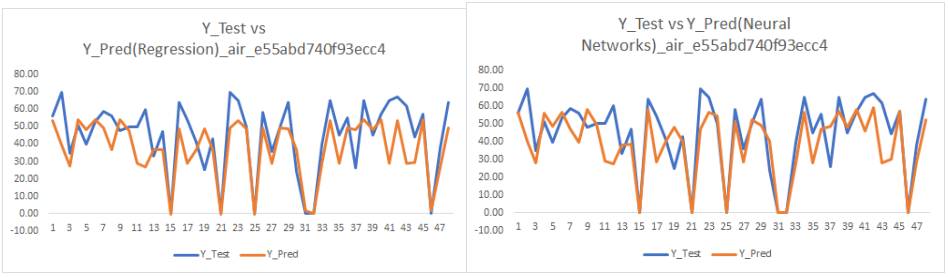
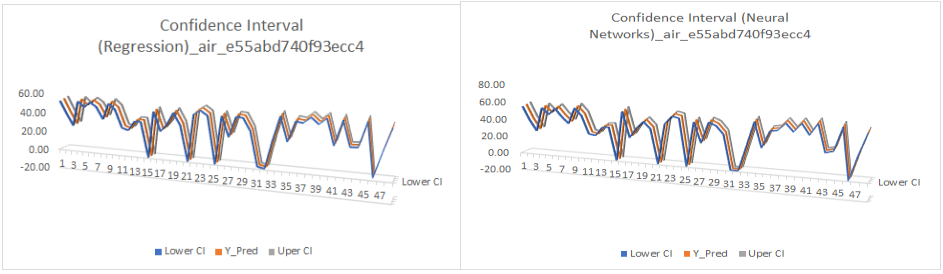


Exhibit 37: 95% confidence interval of Prediction



Business Interpretation: Predicting the number of visitors for restaurants based on location will help business owners to plan for the seating and parking arrangements for the restaurant. This will be especially useful in crowded metros like Tokyo for a seamless customer experience.

References

Zerom, D.(2017). Introduction to Business Forecasting.

Bruce,P., Patel, N.R., Shumeli, G. (2007).Data Mining for Business Intelligence: Concepts, Techniques, and Applications in Microsoft Office Excel with XLMiner (3rd ed.).

Hyndman, R., & Athanasopoulos, G. (2013). Forecasting: principles and practice. OTexts.

Kroenke, D., et al. (2017). Database Concepts (8th ed.). Pearson, ISBN: 013460153X.