RISE BY DIGITALBCG ACADEMY

# Capstone Project

Presented by: BDA09 Group05 - The Rising Seven


13 JAN 2022

# How might we provide better allocation of buses to bus stops to cater to ridership demand?

## Our Recommendations

### Data

Adopt a demand forecasting model to help Sentosa Development Corporation (SDC) to preempt the hourly ridership a week ahead to allow SDC Operations team to plan its bus scheduling to cater to the guest demand.

### Digital

Include shuttle bus service information in the Sentosa App to allow guests to view the bus arrival times and available capacity real time.

## Value & Impact

1) More dynamic scheduling ahead of demand surges instead of current fixated schedules.
2) Potential cost savings to remove under-utilized buses during non-peak hours.
3) Increase in customer satisfaction as they can use the App to better plan their time in Sentosa
4) Potential increase in revenue for SDC with more take up traffic on the Sentosa App.

# Project Modelling Flow

**Business Understanding & frame the problem Statement/Hypothesis**

**"How might we provide better allocation of buses to bus stops to cater to ridership demand?"**

**Initial Data Understanding & Processing**

**- Obtain dataset**
**- Understand and clean data**

**Data Analysis**

**Gather insights from initial data:**
- Peak of day
- Highest ridership by bus stop/bus route
- Any correlation between ridership and weather, weekends, etc.

**Modelling**

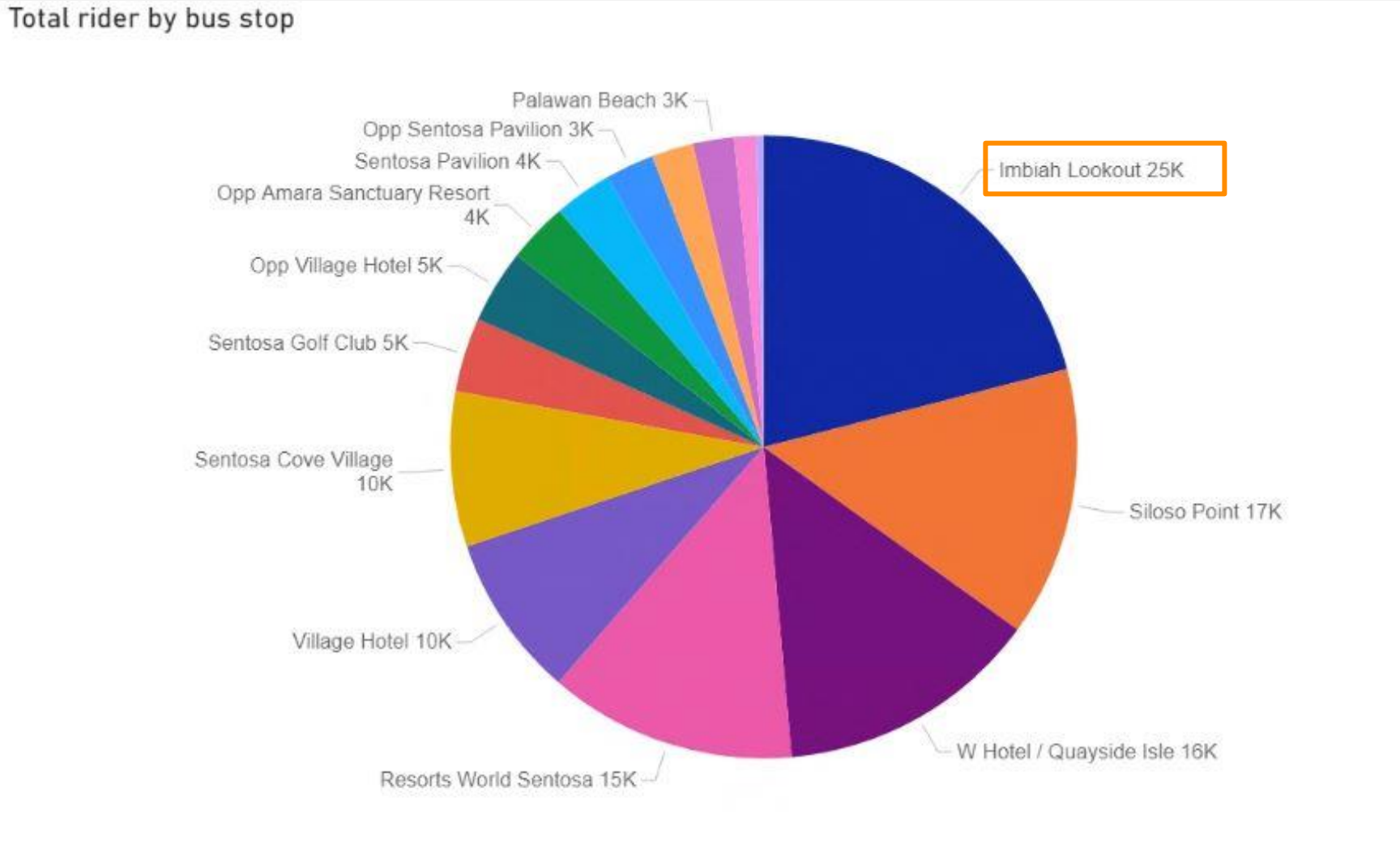
**Create new features to improve predictions:**
- Average ridership
- Bus frequency

**Built a model to predict hourly ridership demand one week in advance**

# Key Assumptions/Limitations

1. Total rider coming in and out are the same number.
2. Bus stops and bus routes cannot be removed.
3. No key events or promotions during the period of data provided (Jan – Mar 2021) due to COVID-19 SMM measures.
4. Maximum bus capacity is 50 pax due to COVID-19 SMM measures.

# Imbiah Lookout has highest total ridership for all Bus Stops



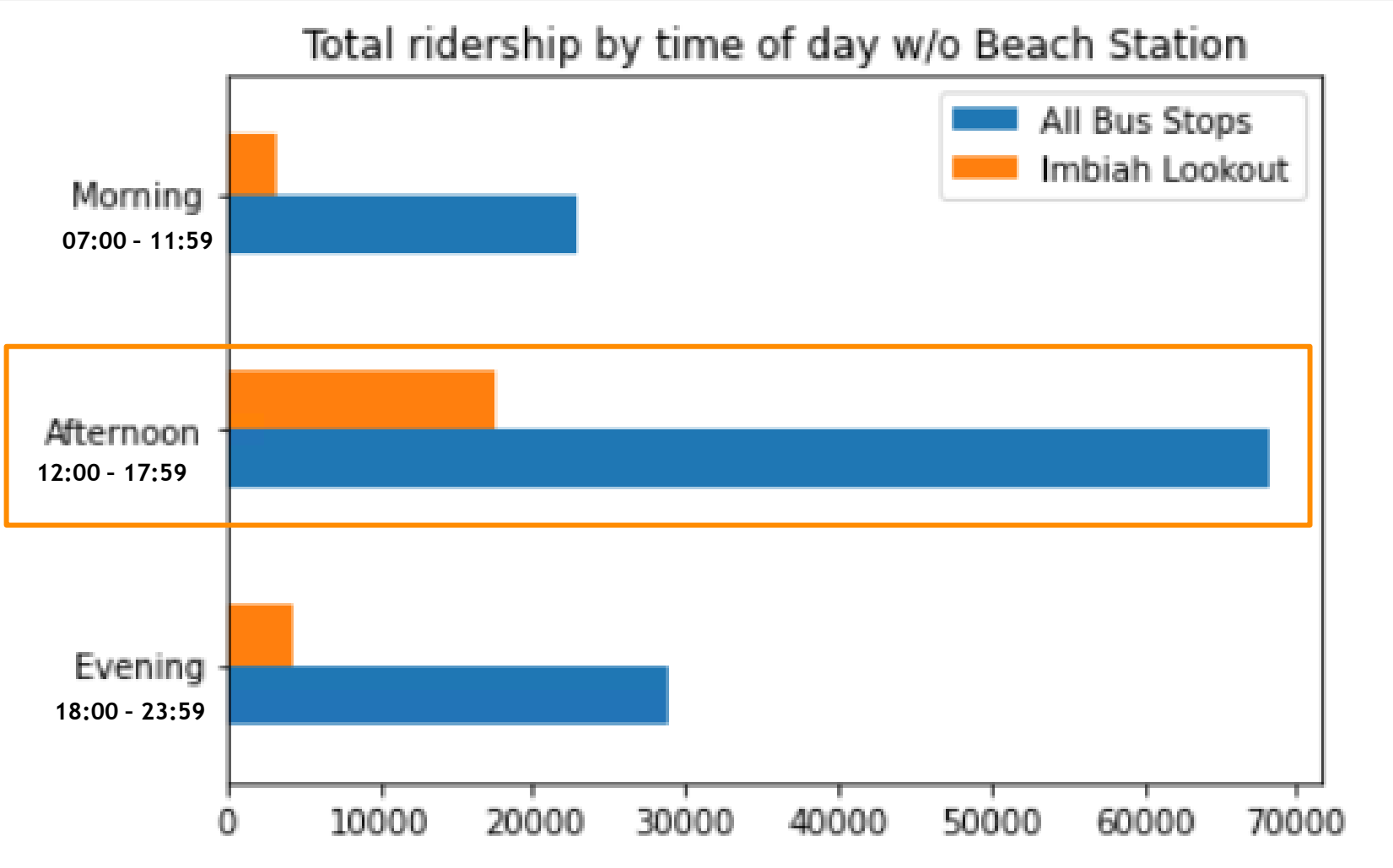

Imbiah lookout has the highest ridership of all bus stops.

**Beach Station is excluded in this analysis as it is a bus interchange which naturally will have high ridership.*

*Our focus is on the bus stops between the interchange.*

Source : 3 months actual ridership data (Jan to March 2021) from SDC

# Afternoon is the peak for all buses (including Imbiah)

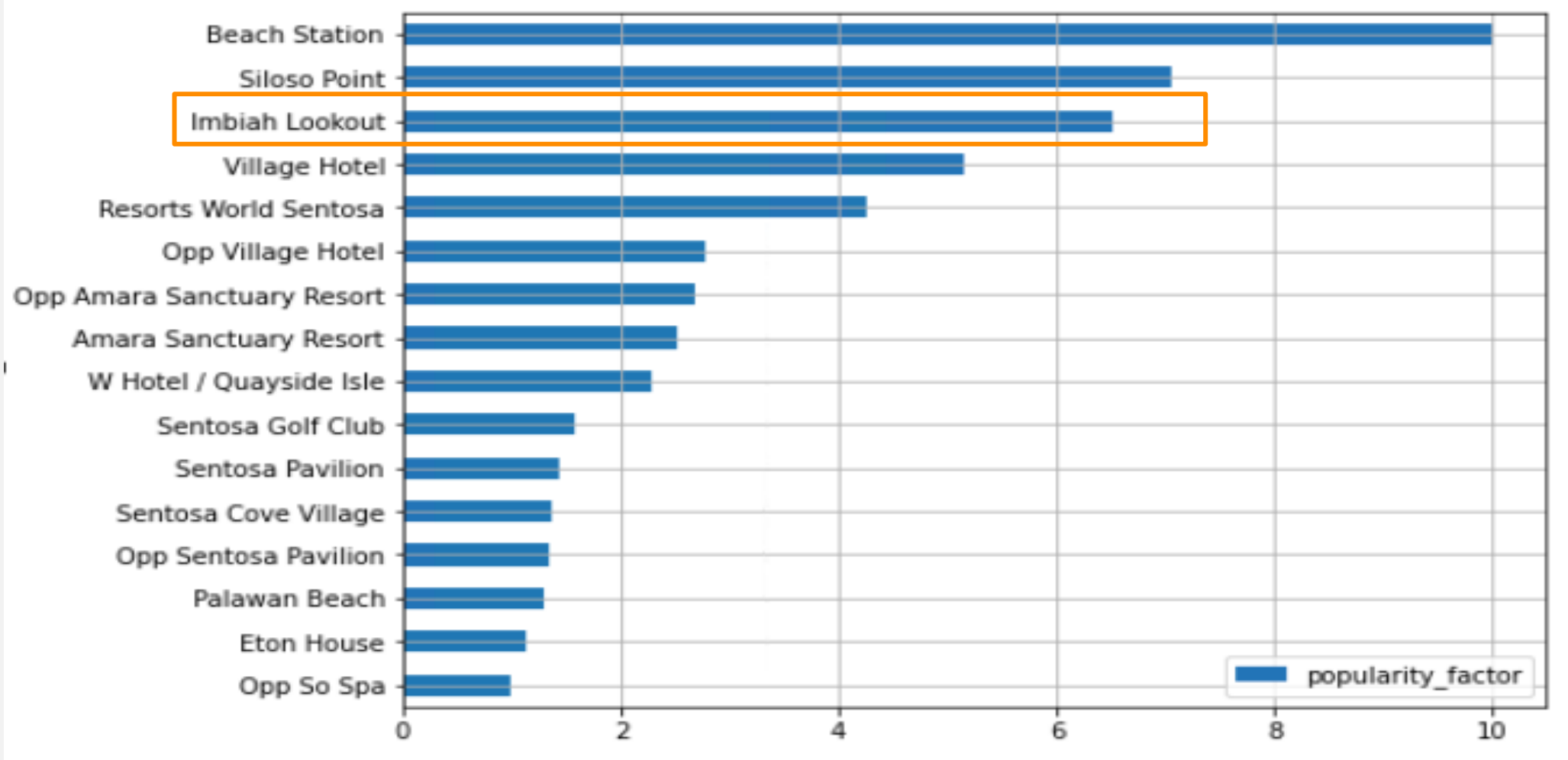



Sentosa visitors generally prefer to take the bus in the afternoon compared to other times of the day.

This suggests that most of the bus congestion happens in the afternoon.

Source: 3 months actual ridership data (Jan to March 2021) from SDC

# Imbiah Lookout is in the top 3 most popular bus stops

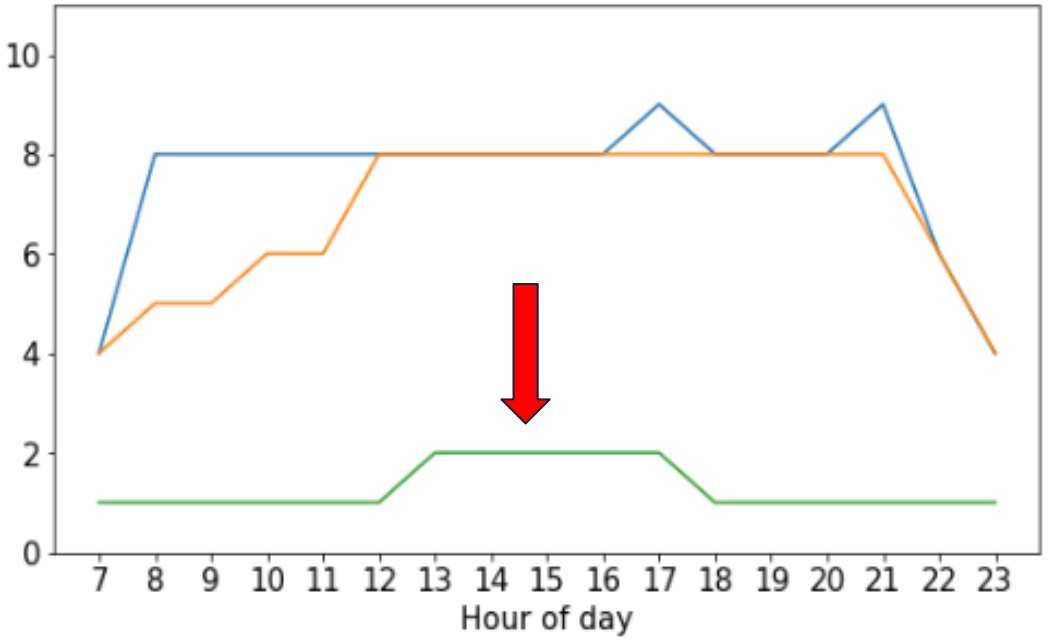



Source : 3 months actual ridership data (Jan to March 2021) from SDC

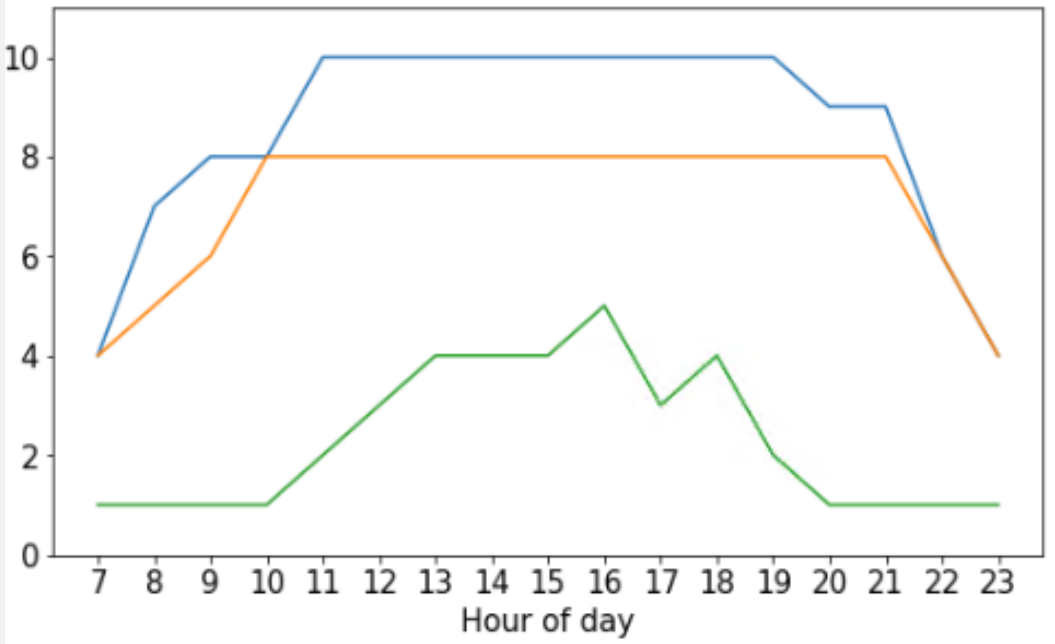
*Refer to appendix for more details on popularity index

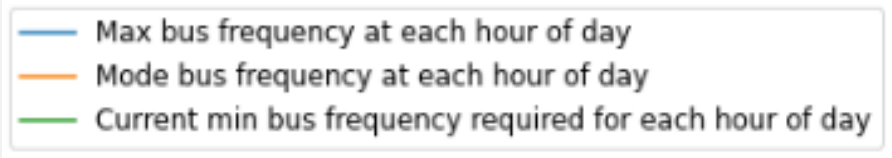
# Bus Frequency of Imbiah Lookout

## Weekdays




## Weekends and holidays

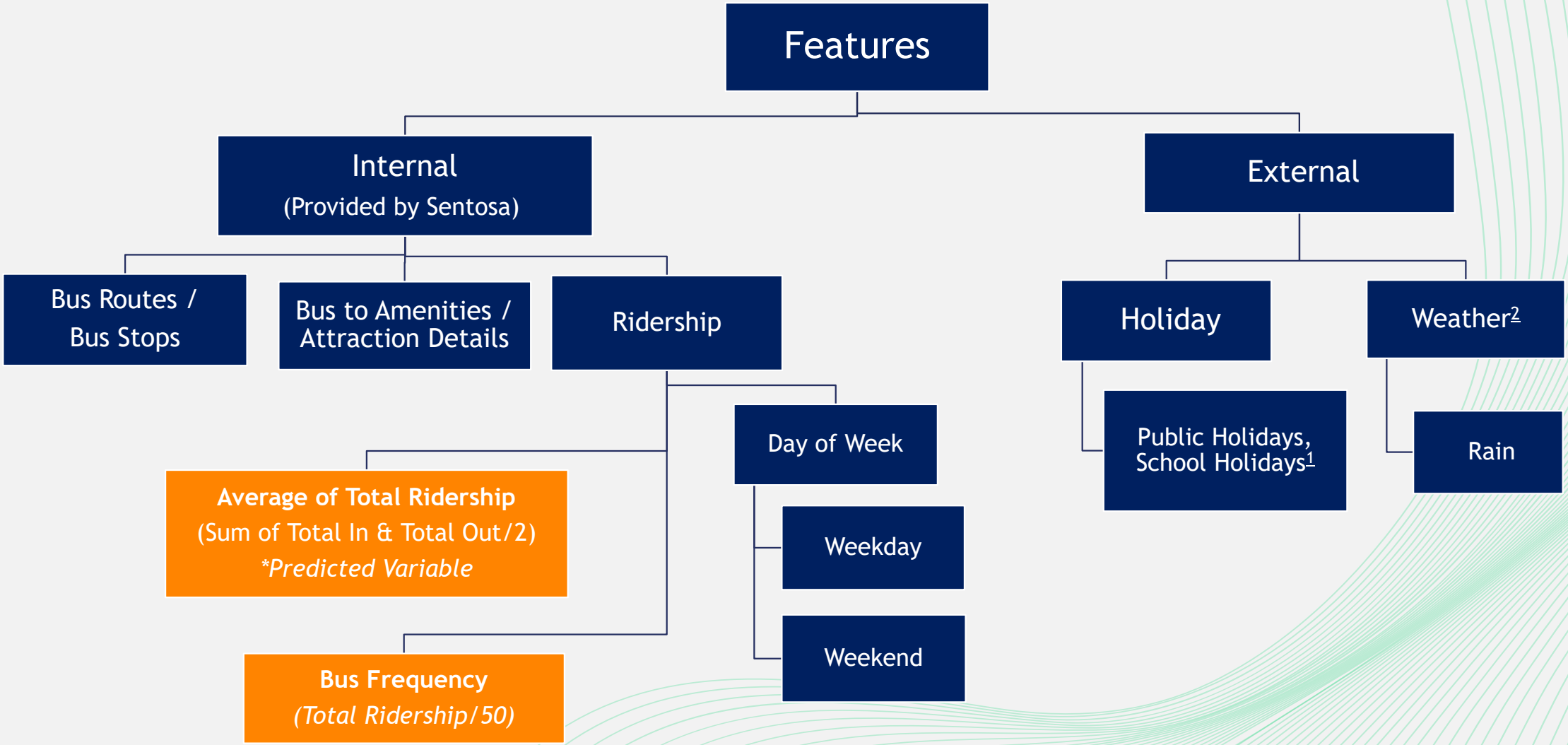



Source : 3 months actual ridership data (Jan to March 2021) from SDC

Based on the highest ridership for every hour and taking the maximum bus capacity of 50pax, we identified that the actual bus deployment required to support the highest ridership could be more than required. There is a possibility of under-utilised buses using the current fixed bus schedule.

# Feature Selection

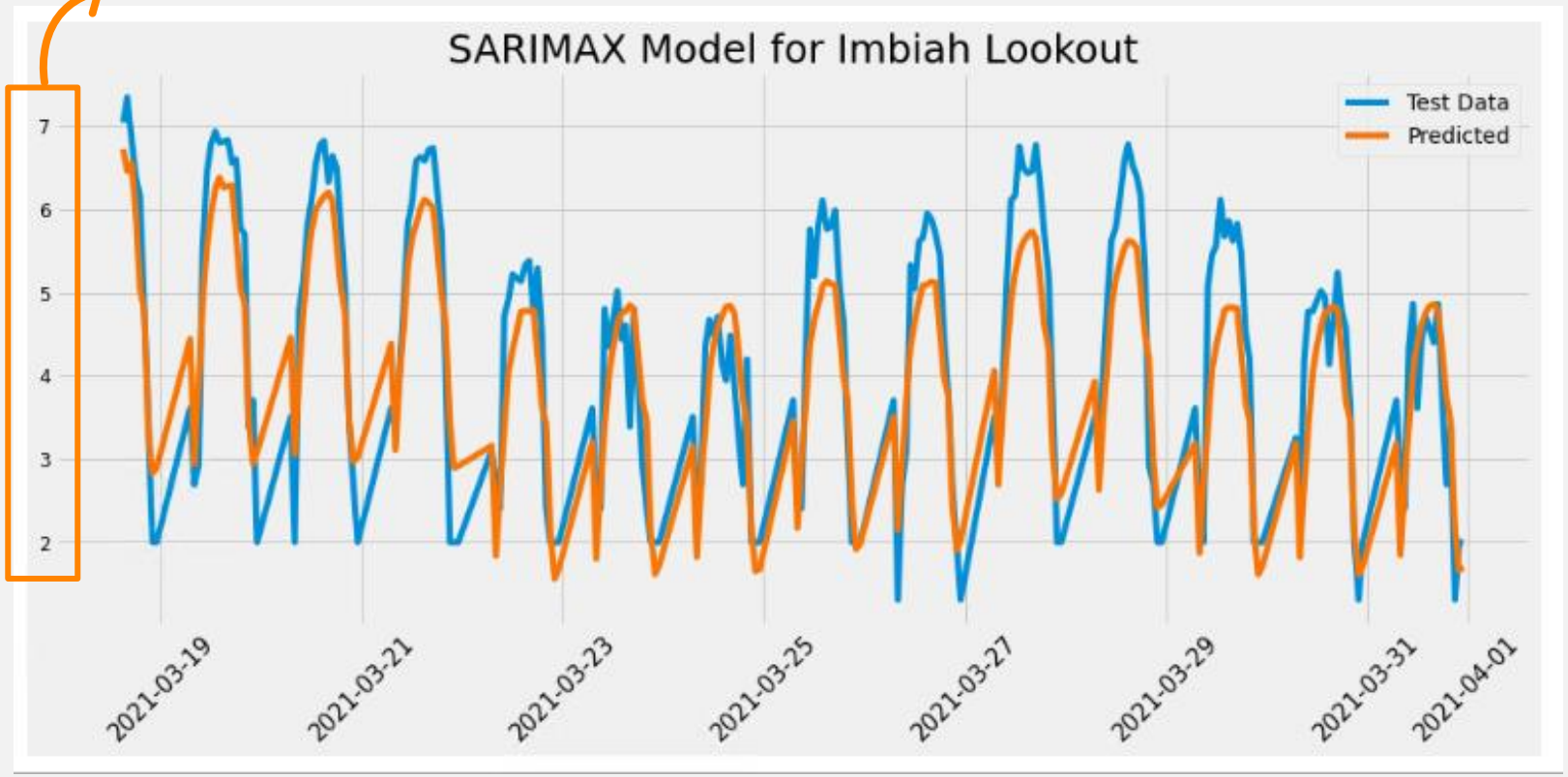





*Source: 1 www.moe.gov.sg/news/press-releases/20200817-school-terms-and-holidays-for-2021; 2 www.weather.gov.sg*

# Time Series Modelling (SARIMAX)

We took a log of the target variable for scaling




| r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|
| 0.8 | 0.83 | 0.61 | 0.52 | 16.25 | 0.72 |

Train/Test Split: 0.85/0.15

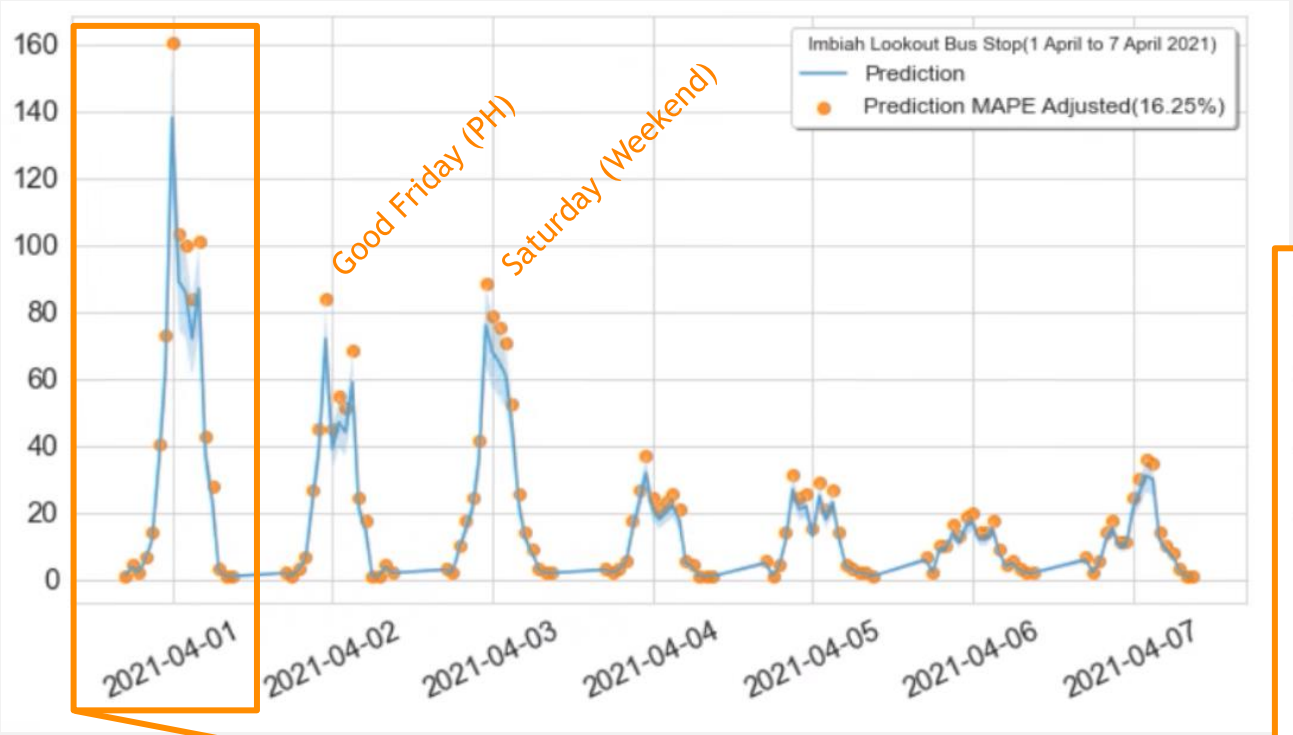
SARIMAX Model was applied to account for seasonal trends and exogenous factors.

Rain, holidays and weekends stood out in the modelling process.

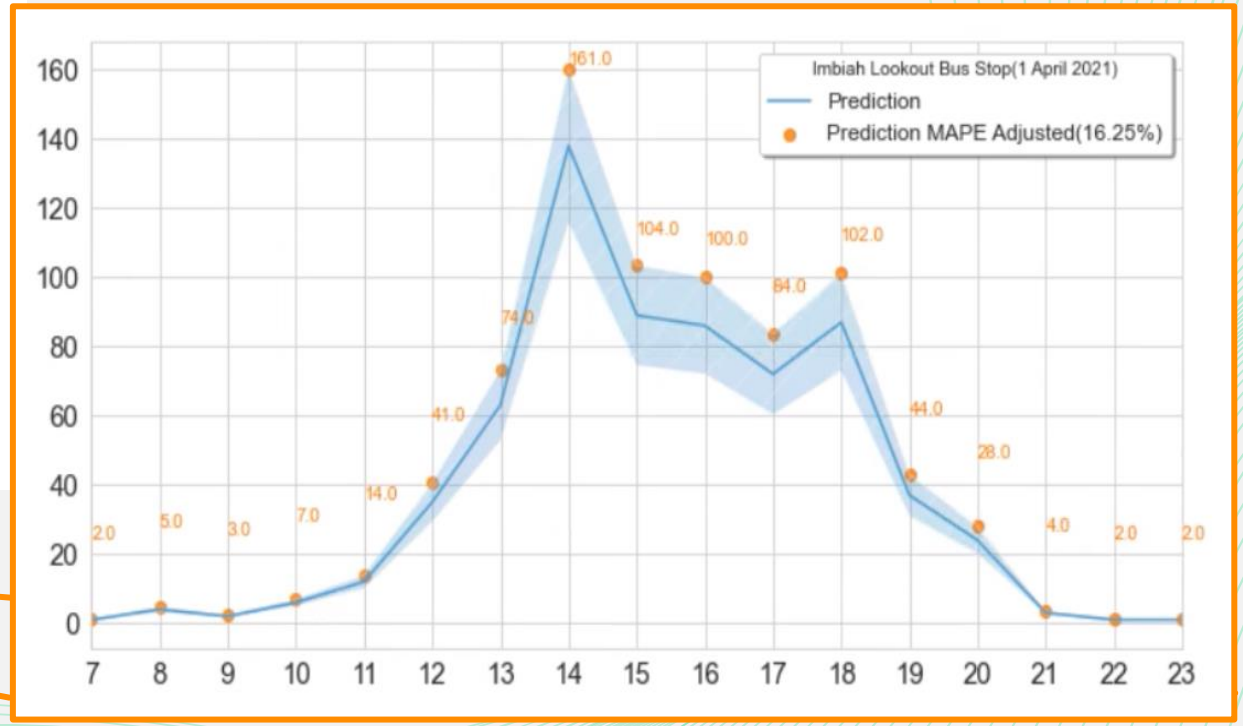
R2 score and Mean Average Percentage Error (MAPE) to determine accuracy of our model.

Source : 3 months actual ridership data (Jan to March 2021) from SDC

# Ridership Predictions for 1st Week of April 2021



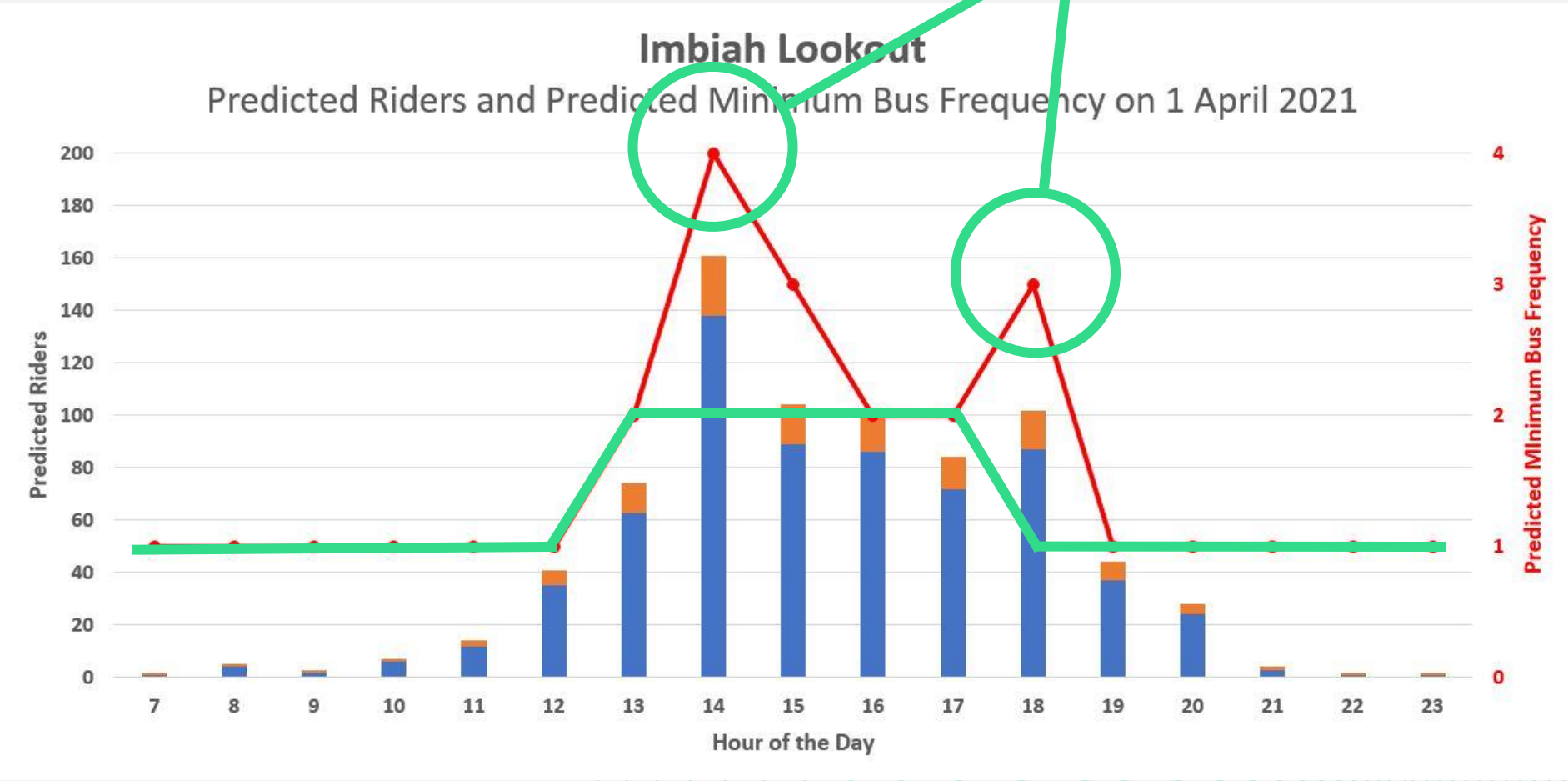

Weekly Predictions




Hourly Prediction for 1 Day

# Predicted Minimum Bus Frequency Required on 1st April 2021

Difference in minimum required bus frequency due to the peaks in demand



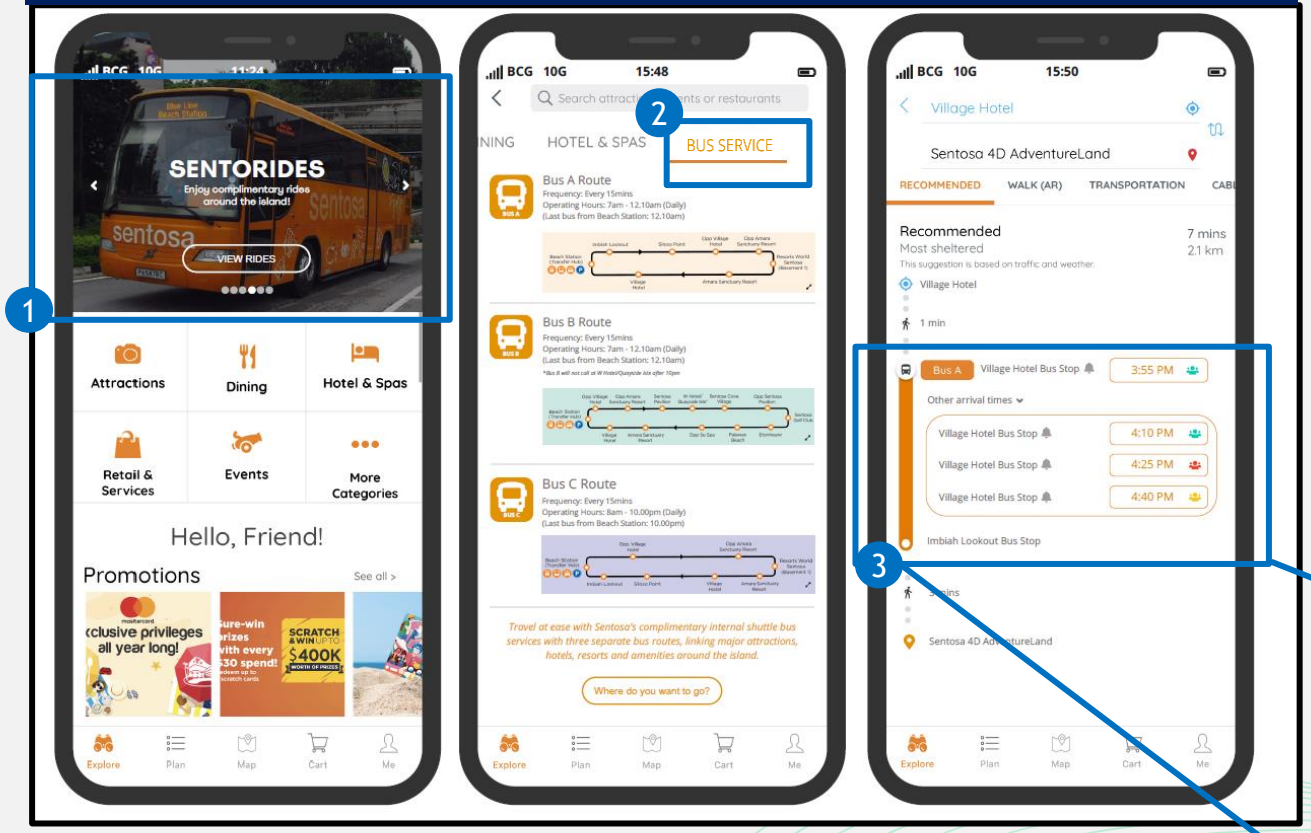

## Legend

- Predicted Ridership
- Additional buffer of **16.25%** according to MAPE Value in Prediction Model
- Predicted Minimum Bus Frequency
- Actual Weekday Minimum Bus Frequency Required (Derived from EDA)

# Adding new features in existing Sentosa App

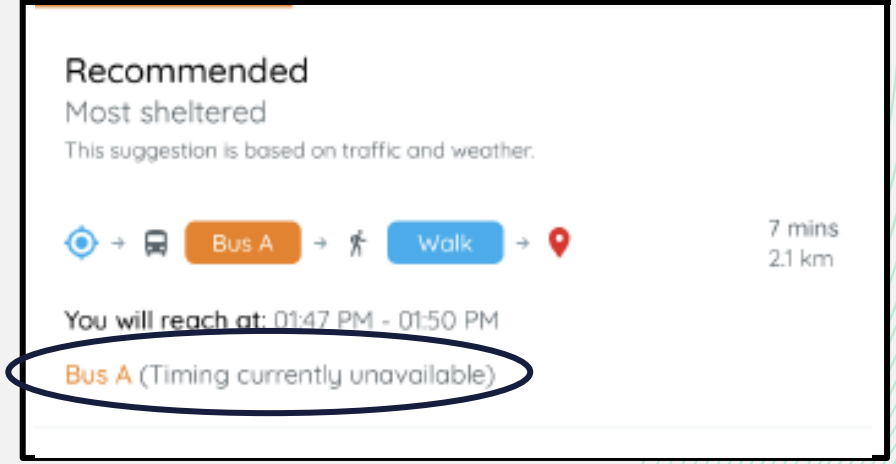
## Shuttle bus services, real time bus arrival and its occupancy

### Proposed Wireframes

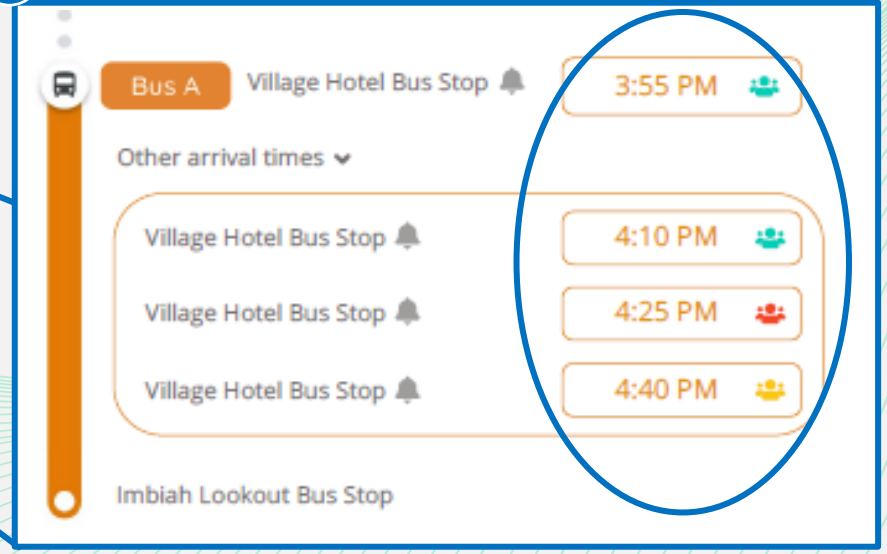



1. Include Sentosa Shuttle bus Services
2. Include additional info page on the available routes
3. Include breakdown of bus arrival time and bus capacity

### Current Design - Missing time of arrival & available capacity




### 3 Proposed Design - Real time bus arrival & available capacity



# Next steps

- Acquire longer period data
- Consider additional features
- Improve & Optimise Model (other bus stops)

Immediate

- Feedback on Model (CSV) and integrate into Dashboard prototype
- User Testing on App

3 Months

Live on App and Dashboard

6 Months

2 Months

- Model Implementation (CSV)
- Develop Dashboard prototype
- Build App Features prototype

4 - 5 Months

- AB Testing on Dashboard & user feedback
- App Soft Launch

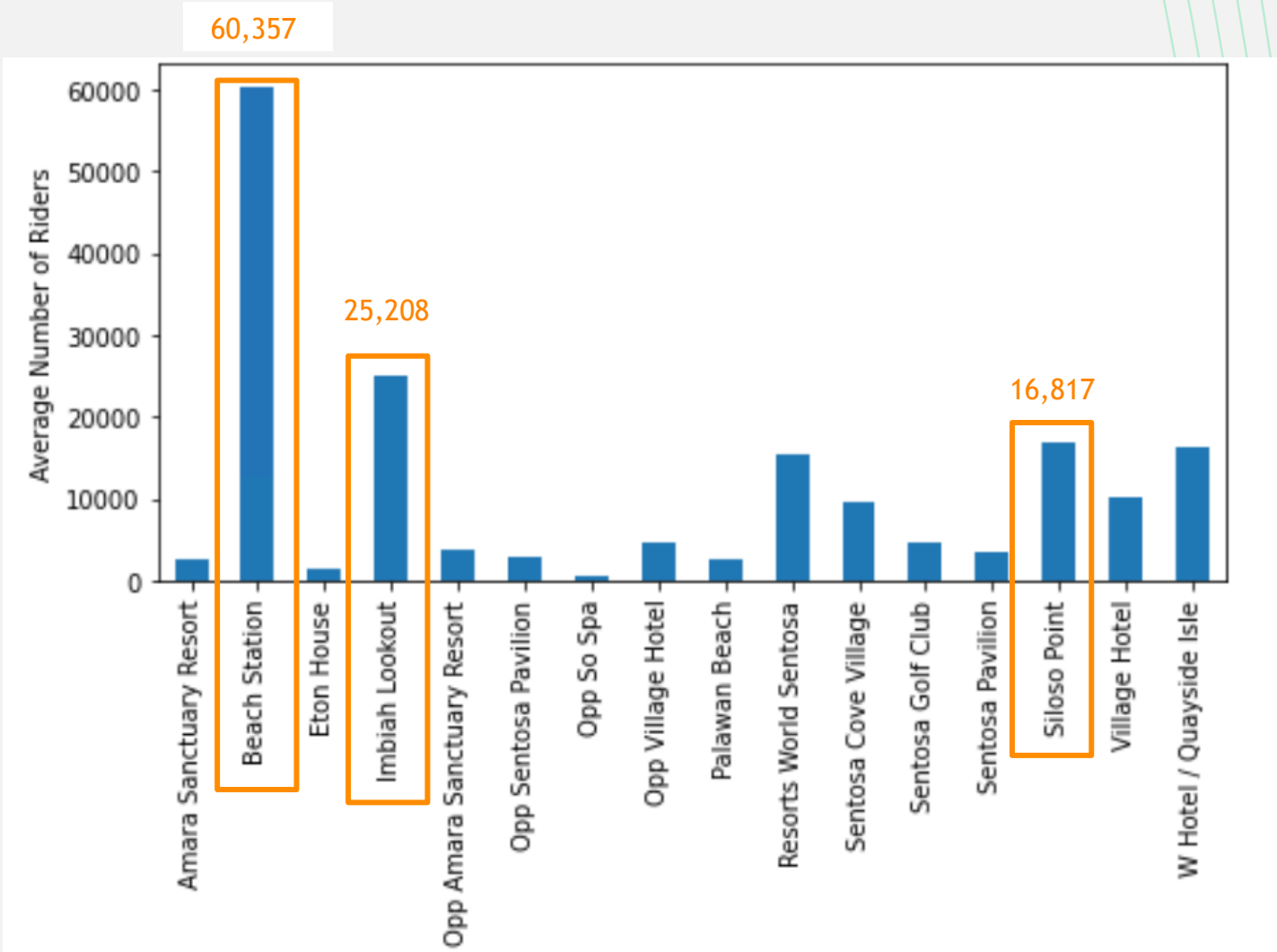
# Q & A

# Appendix

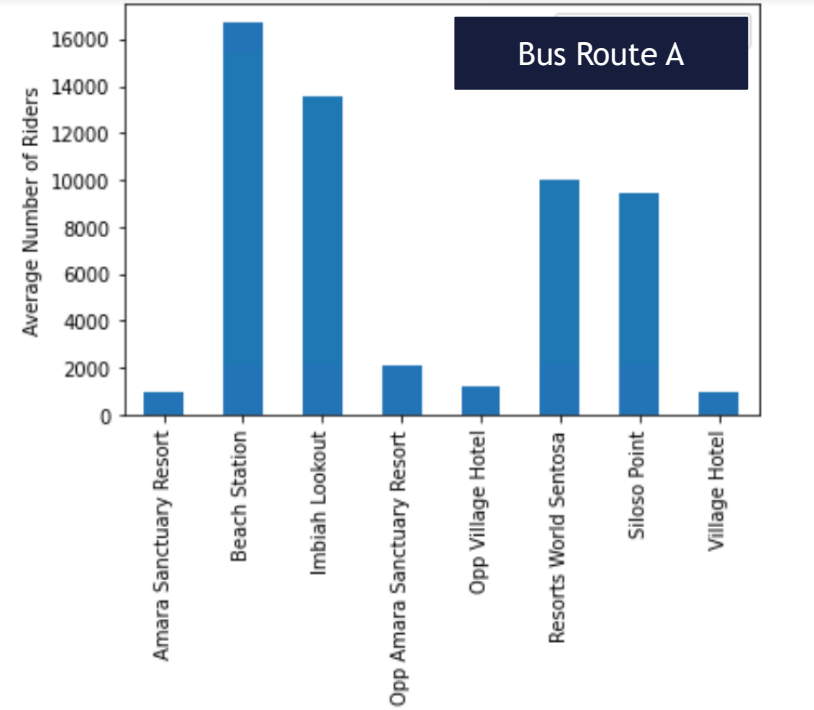
# Total ridership for bus stops (including Beach station)

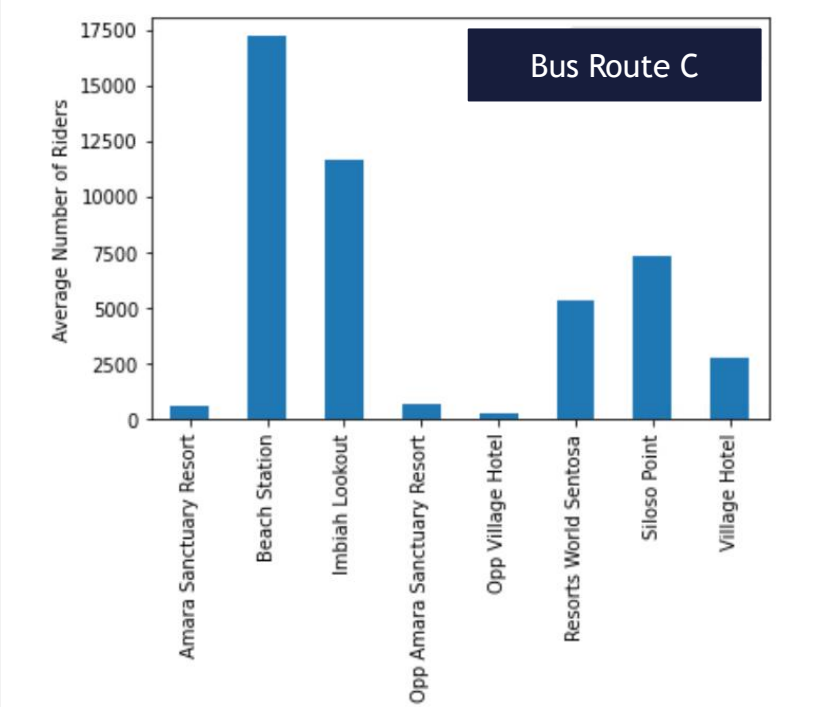
## Analysing average number of riders against bus stops

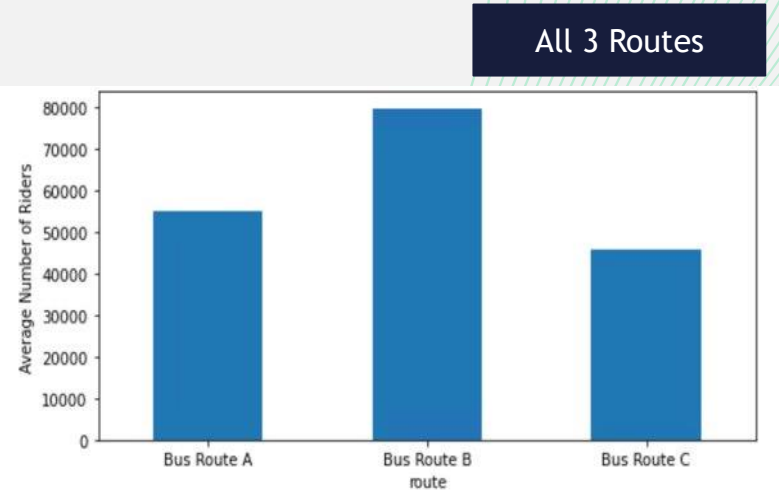
| bus_stop | total_riders/ |
|---|---|
| Beach Station | 60357. |
| Imbiah Lookout | 25208. |
| Siloso Point | 16817. |
| W Hotel / Quayside Isle | 16346. |
| Resorts World Sentosa | 15374. |
| Village Hotel | 10157. |
| Sentosa Cove Village | 9674. |
| Sentosa Golf Club | 4626. |
| Opp Village Hotel | 4588. |
| Opp Amara Sanctuary Resort | 3778. |
| Sentosa Pavilion | 3602. |
| Opp Sentosa Pavilion | 3030. |
| Amara Sanctuary Resort | 2585. |
| Palawan Beach | 2549. |
| Eton House | 1370. |
| Opp So Spa | 450. |



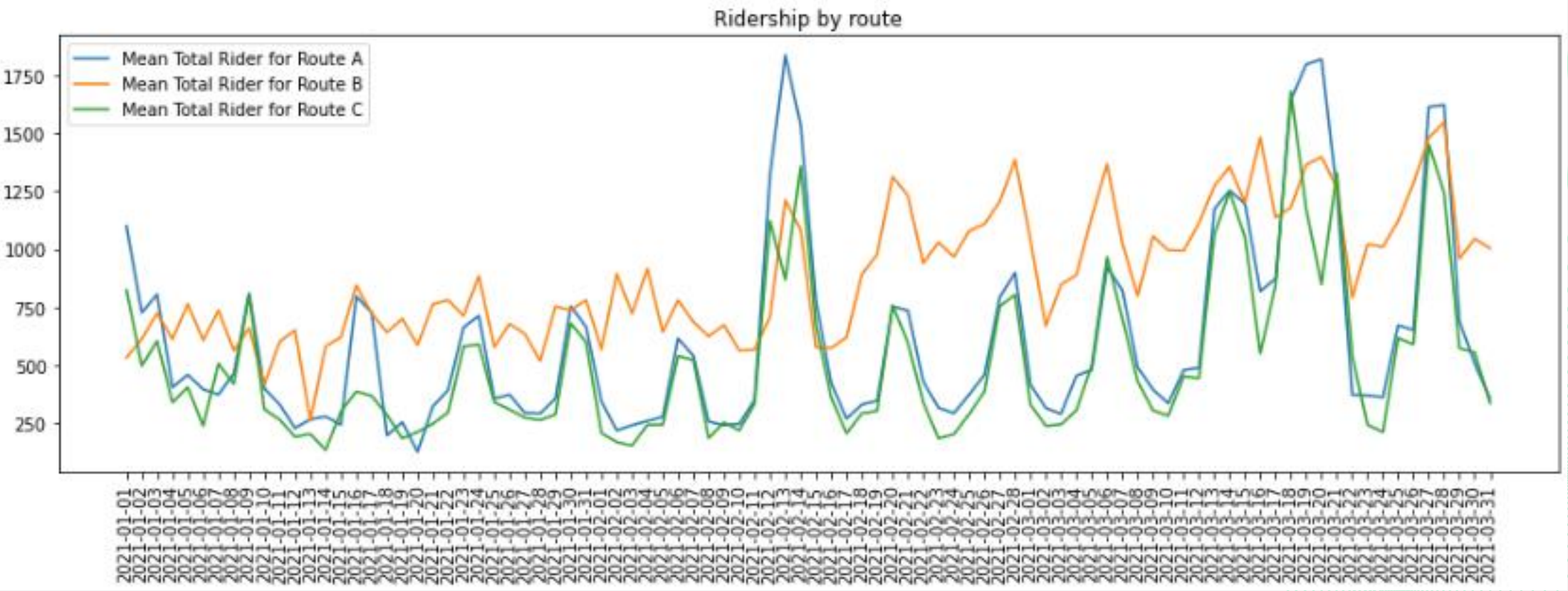
# Imbiah Lookout and Siloso point are the top 2 most popular bus stops (ridership)



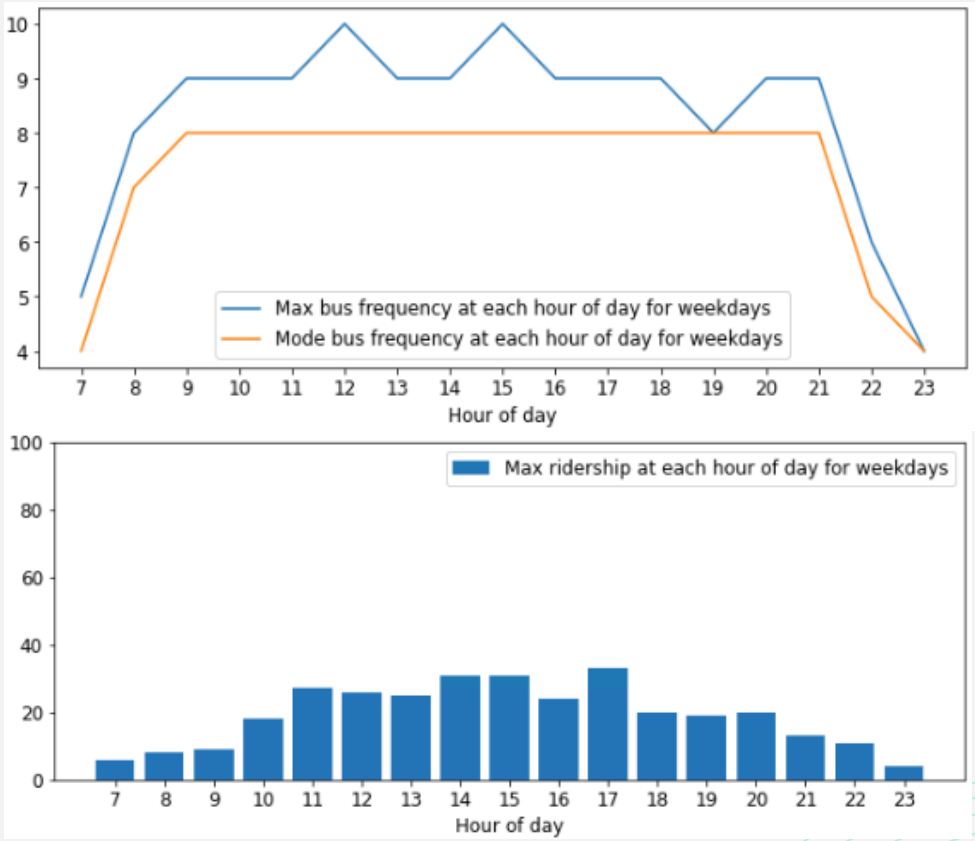



# Total ridership by route

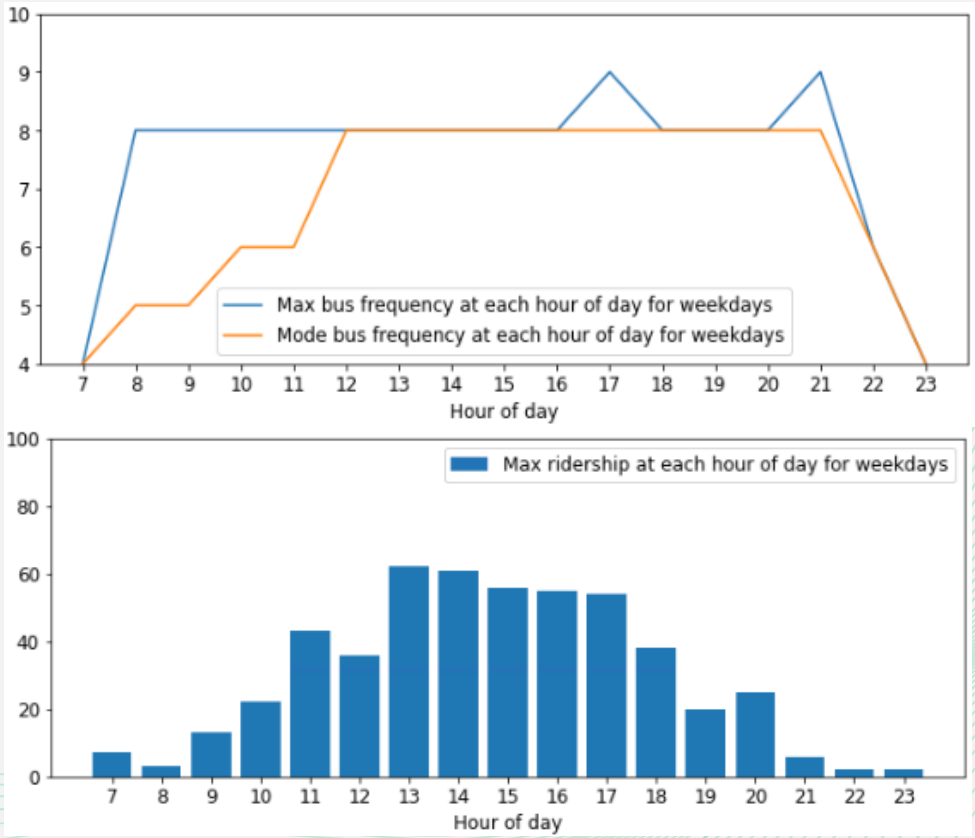


# Bus Frequency of top bus stops
# Siloso Point and Imbiah Lookout (Weekday)

## Siloso Point



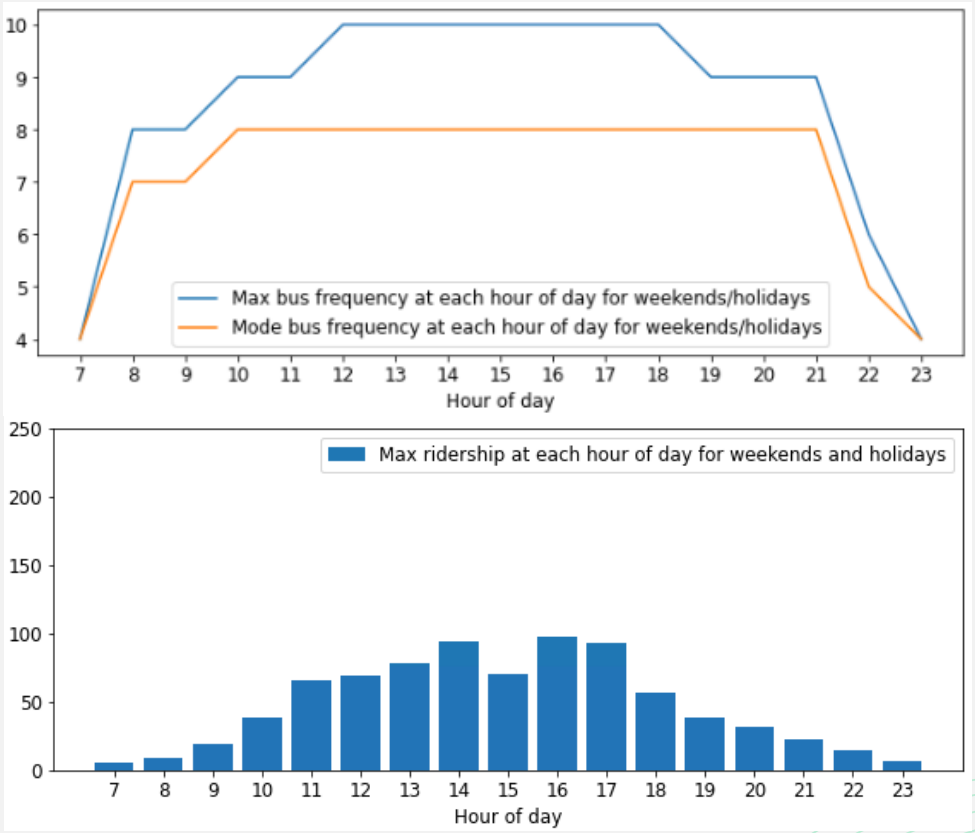

## Imbiah Lookout



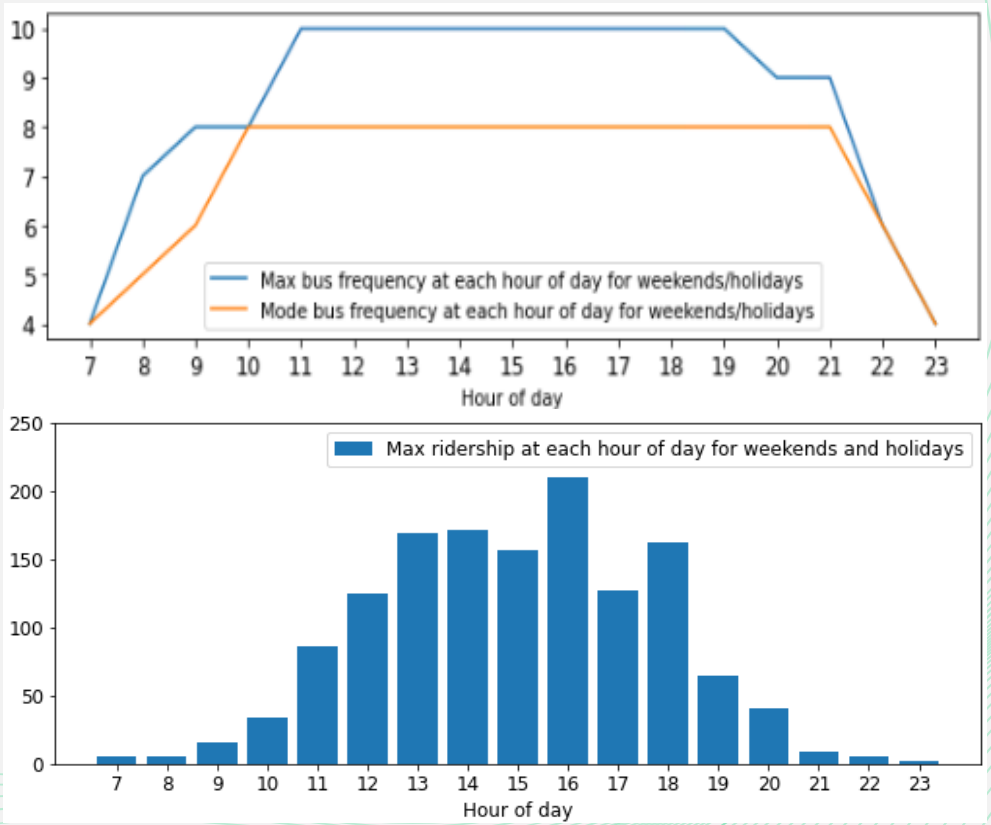

Based on maximum bus ridership within each hour and bus capacity of 50, actual frequency of buses deployed within each hour on weekdays is higher than required.

# Bus Frequency of top bus stops
# Siloso Point and Imbiah Lookout (Weekend/holidays)

## Siloso Point

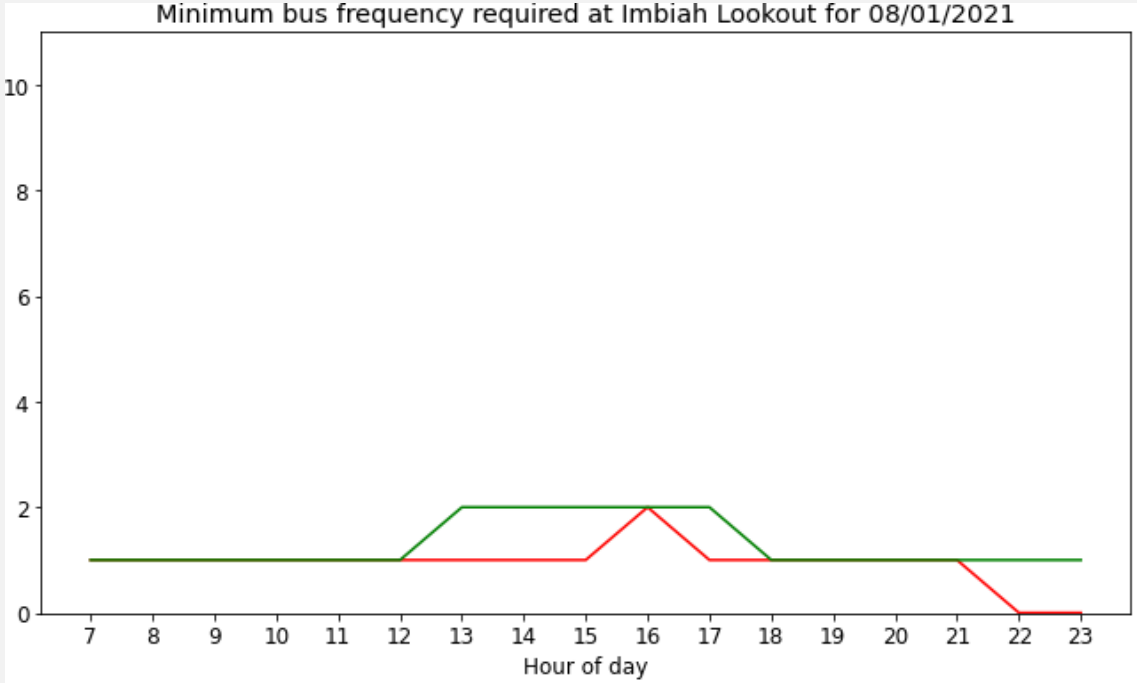


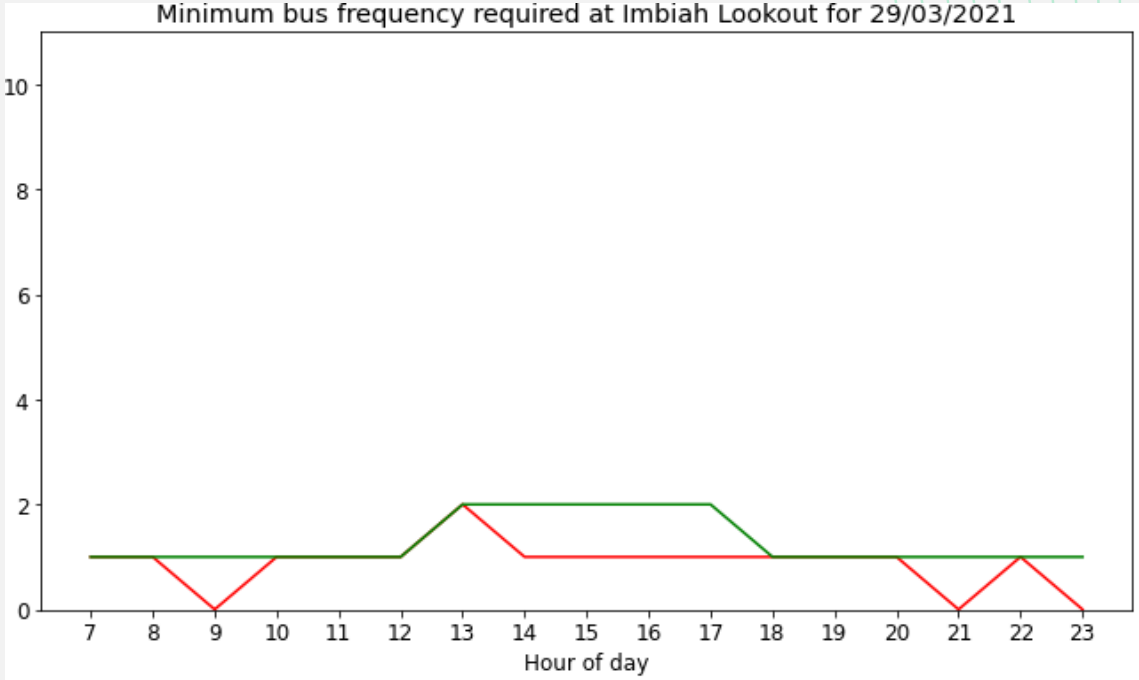

## Imbiah Lookout




Based on maximum bus ridership within each hour and bus capacity of 50, actual frequency of buses deployed within each hour on weekends is higher than required.

# Comparing Minimum Bus Frequency Required:
## Selected Weekday vs 3 Months of Weekdays Data

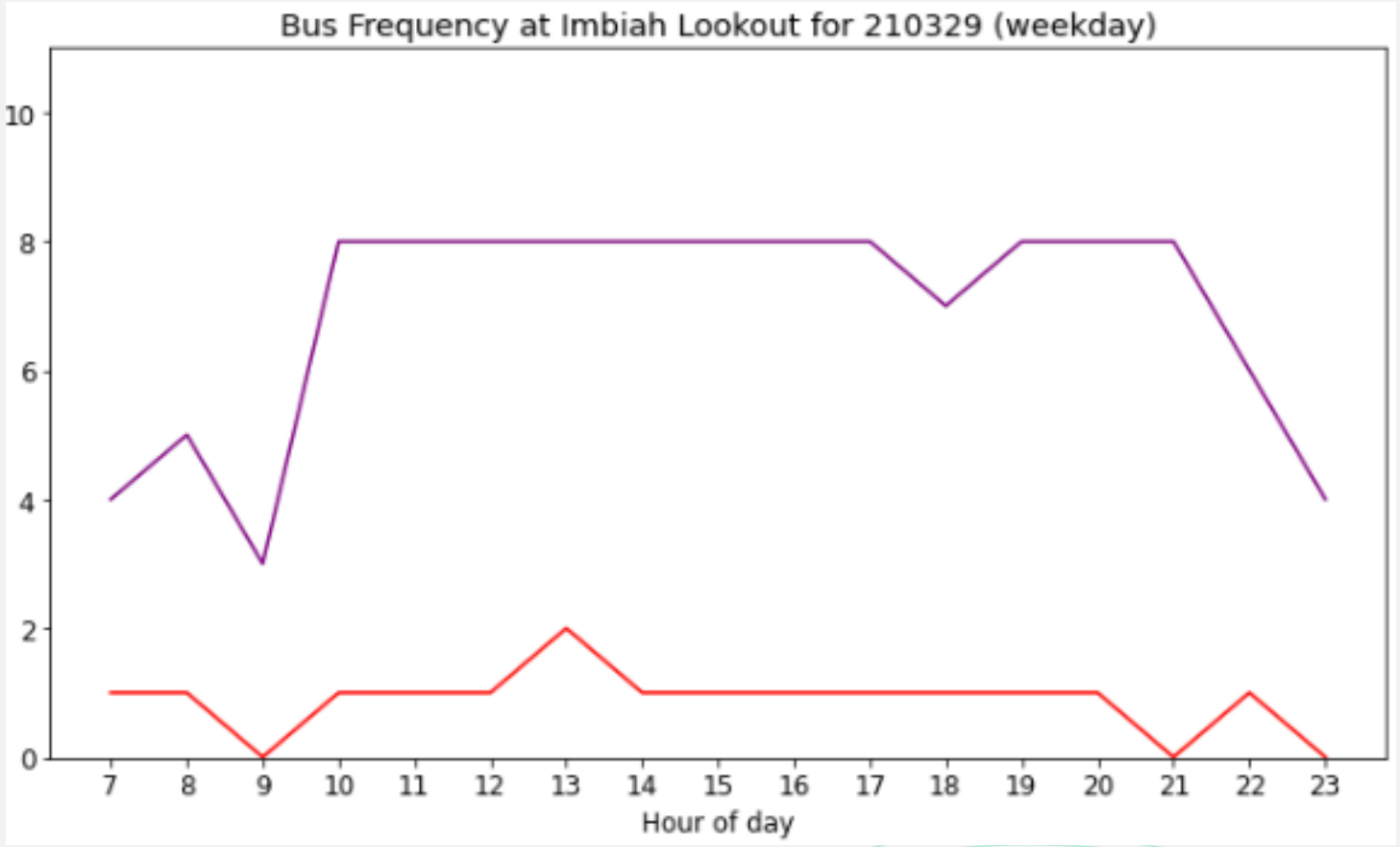


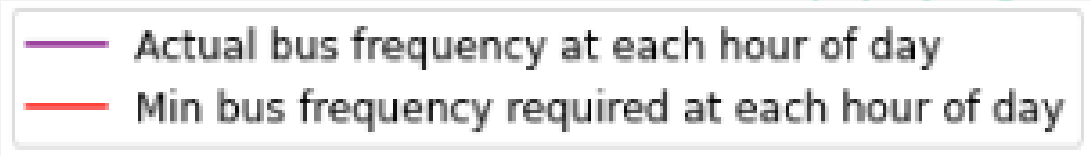




Min bus frequency required at each hour of day for selected weekday

Min bus frequency required at each hour of day for max ridership of whole period

# Select Weekday Bus Frequency Comparison:
## Actual Bus Deployed vs Minimum Bus Frequency Required



# Calculation for Popularity Factor

```
#Adjusting popularity factor to put more weightage on attractions
#Insert score weightage here
attraction = 0.8
fnb = 0.1
hotel = 0.05
transport = 0.05
```

| Bus Stops | Lat | Long | Route Sequence | Attractions/ Landmarks nearby | Amenities nearby | Nearby F&B | Remarks |
|---|---|---|---|---|---|---|---|
| Imbiah Lookout | 1.25548708 | 103.816093 | 2 | 4D Adventureland, Skyline Luge, MegaAdventure, Cable Car (MFLG Line and Sentosa Line), Madame Tussauds + IOS Live, Butterfly Park, Sentosa Nature Discovery/ Geology Gallery, Imbiah Trail | | Starbucks | Some of the Attractions have retail outlets and they sell snacks |

The score for Popularity Factor is calculated with the following weightage.

We put Attraction as the highest percentage as we want to focus on guests more interested in Attractions compared to F&B and Hotels.

# Hypothesis Testing for Correlation between Variables

**Hypothesis testing for yes/no flags**

```
'rain_flag', 'weekend_flag', 'saturday_flag', 'sunday_flag', 'midweek_flag', 'holiday_flag',
   'weekday_holiday_flag', 'weekend_holiday_flag'

Null H0: The correlation between the two variables is zero.
Alte H1: The Correlation between the two variables is not zero.

P-Value = 0.05
If P-value less than 0.05, we reject the Null.  There is a statistically significant correlation
If P-value more than 0.05, we accept the Null.  There is no correlation
```

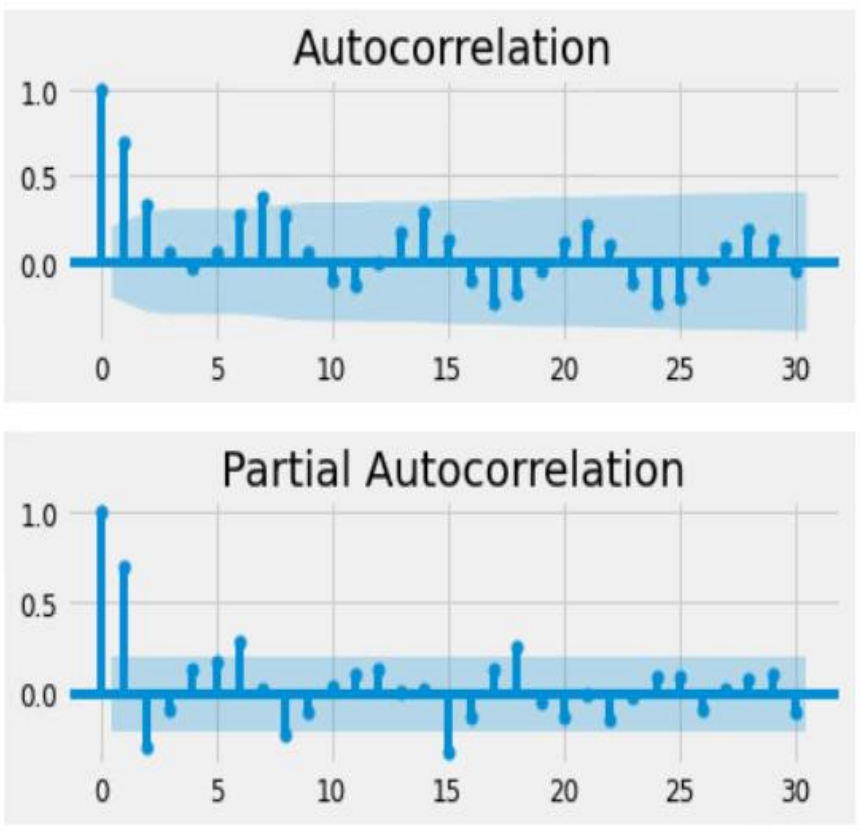
| Mannuwhitneyu test | Rain flag | Weekend flag | Saturday flag | Sunday flag | Midweek flag | Holiday flag | Weekday Holiday flag | Weekend Holiday flag |
|---|---|---|---|---|---|---|---|---|
| P-Value | 0.00 | 0.00 | 0.00 | 0.00 | 0.99 | 0.00 | 0.00 | 0.00 |
| Null H0 | Reject | Reject | Reject | Reject | Accept | Reject | Reject | Reject |

Variable flag and Total Riders are correlated if P-Value is less than 0.05

The variable flag will affect the total ridership if they are correlated.

# **Appendix:** Modelling Test For Daily Prediction

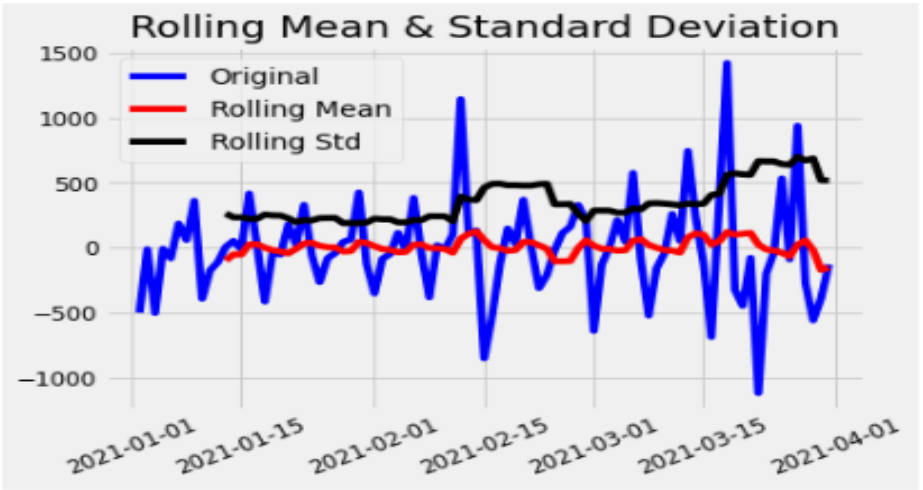
# Preprocessing for Modelling Test for Daily Prediction




TS's ACF & PACF

```
ts_diff.dropna(inplace = True)
test_stationarity(ts_diff)
```

```
ts_diff = ts-ts.shift() #shift all the number in ts next row ahead
plt.plot(ts_diff)
plt.xticks(rotation = 35);
```



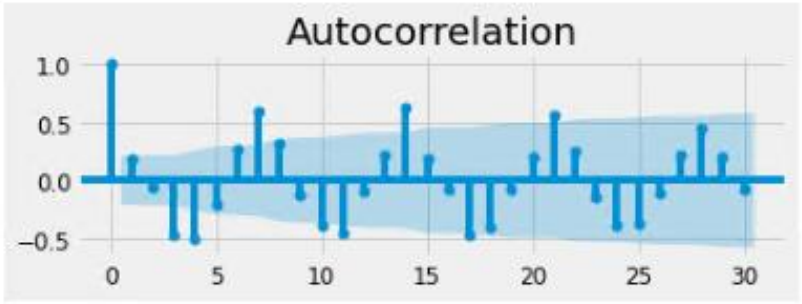

```
Result of Dickey-Fuller Test:
Test Statistic                  -8.406170e+00
p-value                          2.164693e-13
#Lags Used                       4.000000e+00
Number of Observations Used      8.400000e+01
Critical Value(1%)              -3.510712e+00
Critical Value(5%)              -2.896616e+00
Critical Value(10%)             -2.585482e+00
dtype: float64
```

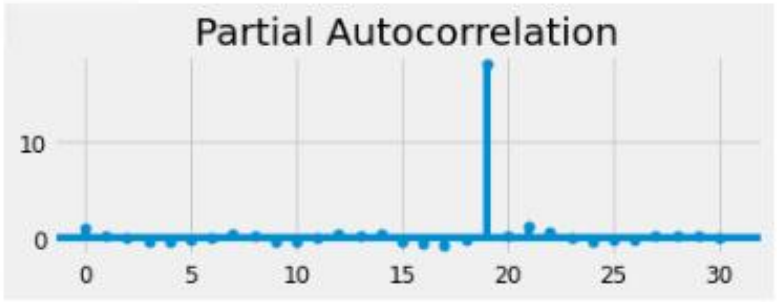
TS with Difference Technique & Stationarity Test

# Preprocessing for Modelling Test for Daily Prediction

```
plt.figure()
plt.subplot(211)
plot_acf(ts_log_diff, ax = plt.gca(), lags = 30)
plt.show()

plt.subplot(212)
plot_pacf(ts_log_diff, ax = plt.gca(), lags = 30)
plt.show()
```

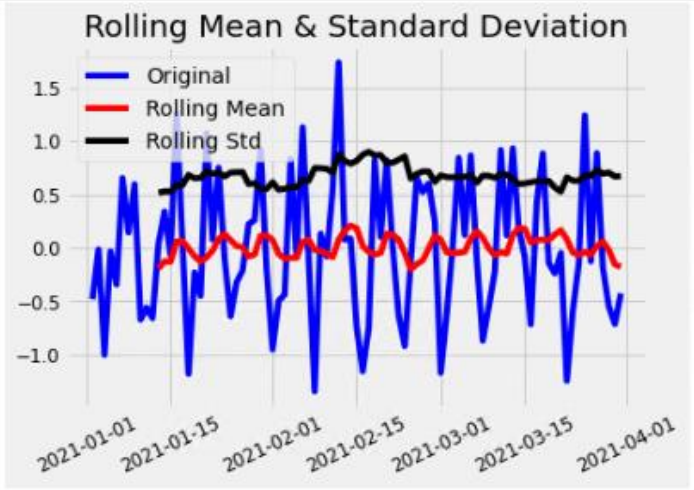



TS Log Diff's ACF & PACF

```
ts_log_diff = ts_log - ts_log.shift()
plt.plot(ts_log_diff)
plt.xticks(rotation = 15);
```

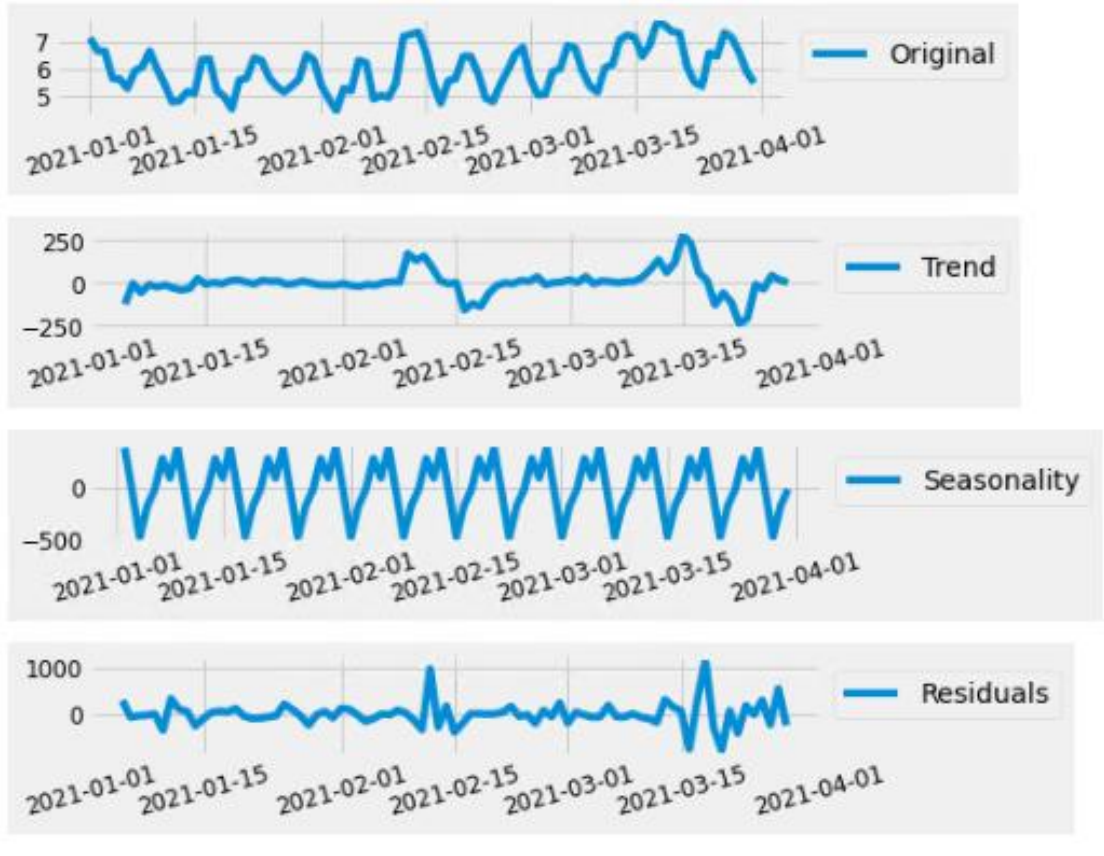
```
ts_log_diff.dropna(inplace = True)
test_stationarity(ts_log_diff)
```



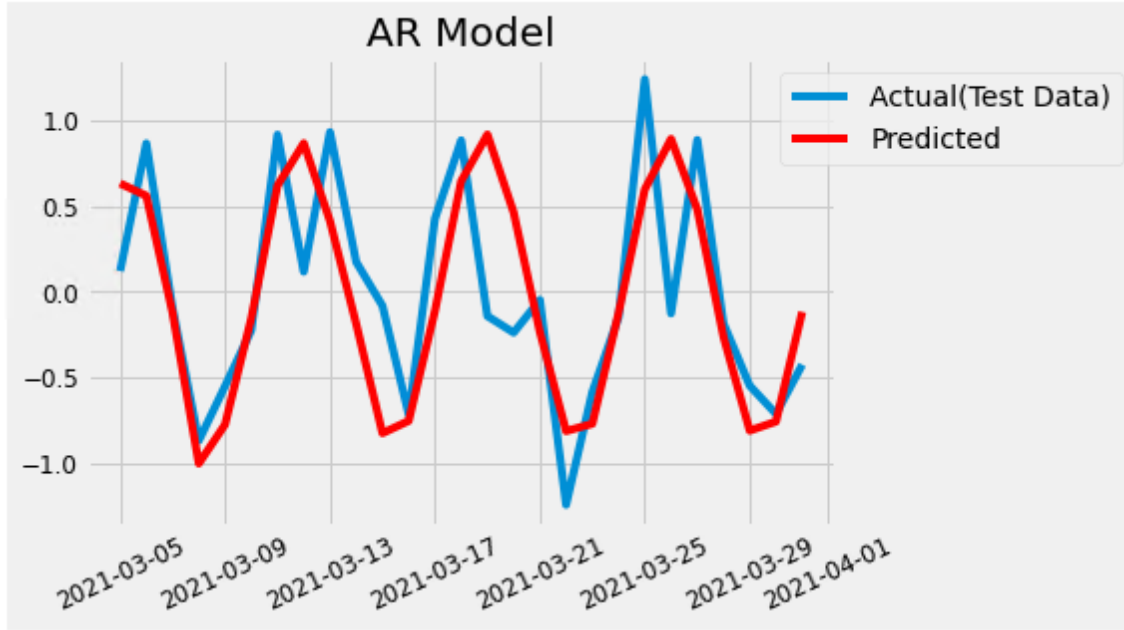

```
Result of Dickey-Fuller Test:
Test Statistic                  -4.782932
p-value                          0.000059
#Lags Used                      10.000000
Number of Observations Used     78.000000
Critical Value(1%)              -3.517114
Critical Value(5%)              -2.899375
Critical Value(10%)             -2.586955
dtype: float64
```

TS Log Transformed with Difference Technique & Stationarity Test

# Modelling Test for Daily Prediction




TS Diff's ACF & PACF




```
evaluate_forecast(test, predictions)
```

| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | 0.42315 | 0.37422 | 0.305855 | 0.223572 | 189.963642 | 0.472835 |

AR Model with TS Diff (No Exog Parameters)

# Modelling Test for Daily Prediction

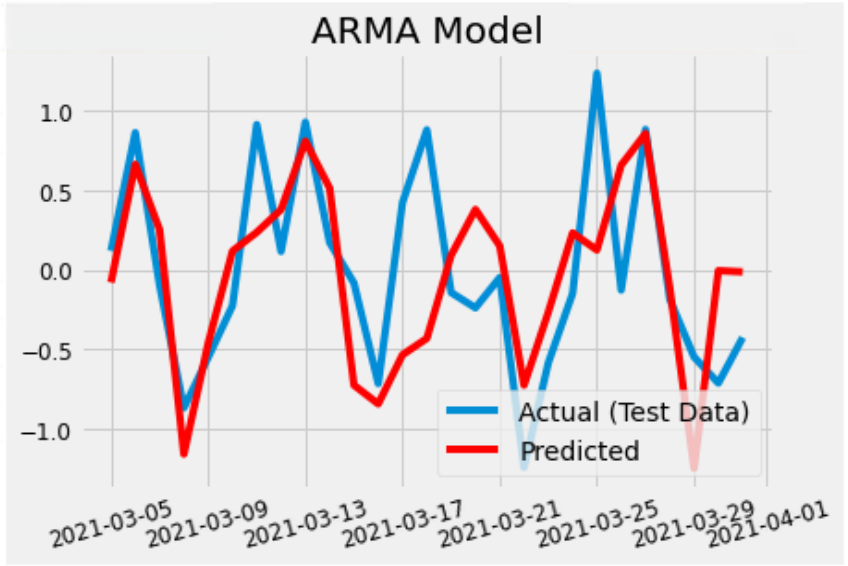
```
exo_train_size = int(len(exo_y) * 0.7)
```

```
exo_train_size
exo_train, exo_test = exo_y[0:exo_train_size], exo_y[exo_train_size:]
exo_train
```

| date | rain_flag | weekend_flag | saturday_flag | sunday_flag | midweek_flag | holiday_flag | weekday_holiday_flag | weekend_holiday_flag |
|---|---|---|---|---|---|---|---|---|
| 2021-01-01 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 2021-01-02 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 2021-01-03 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 2021-01-04 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2021-01-05 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2021-02-27 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 2021-02-28 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 2021-03-01 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2021-03-02 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2021-03-03 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

62 rows × 8 columns

Exog Variables Tabular Format

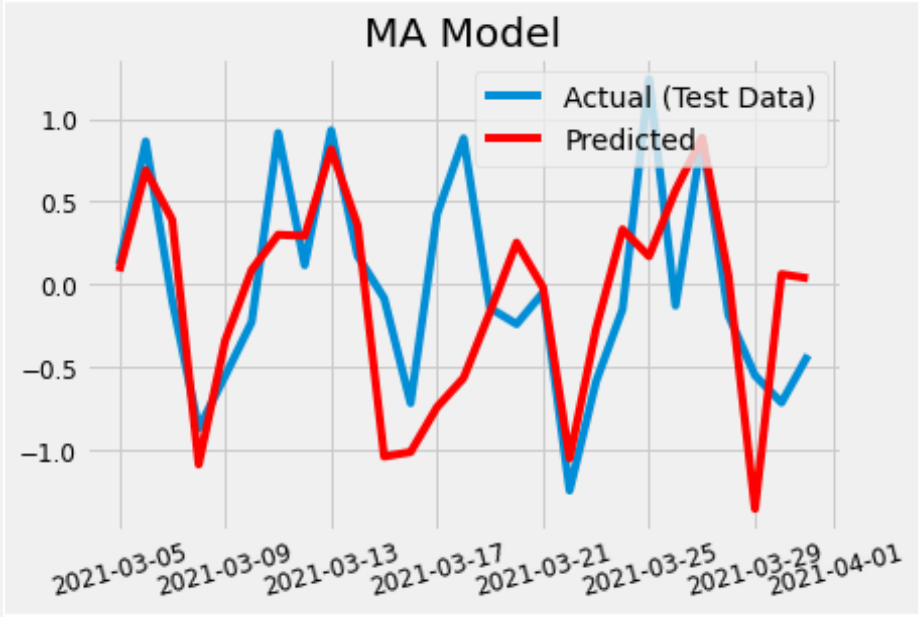



```
evaluate_forecast(test, predictions)
```

| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | 0.217552 | 0.445393 | 0.344517 | 0.303257 | 173.413795 | 0.550687 |

ARMA Model with TS Diff(With Exog Parameters)

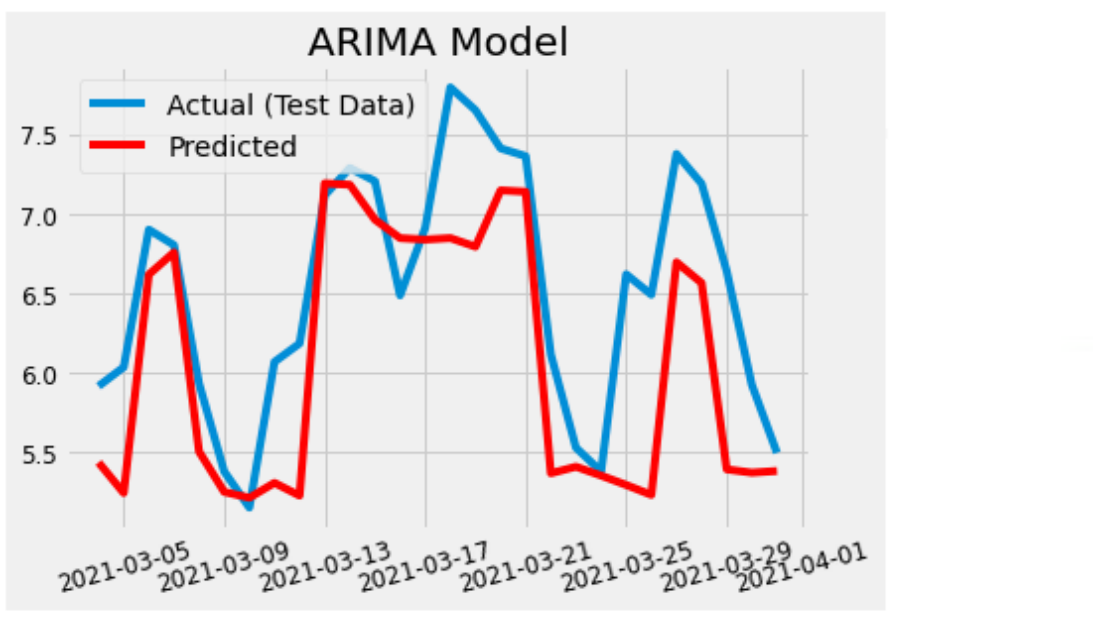
# Modelling Test for Daily Prediction (Moving Average & Arima)




```
evaluate_forecast(test, predictions)
```

| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | 0.1263 | 0.444187 | 0.314097 | 0.338624 | 167.309484 | 0.581914 |

MA Model With TS Diff (With Exog Parameters)

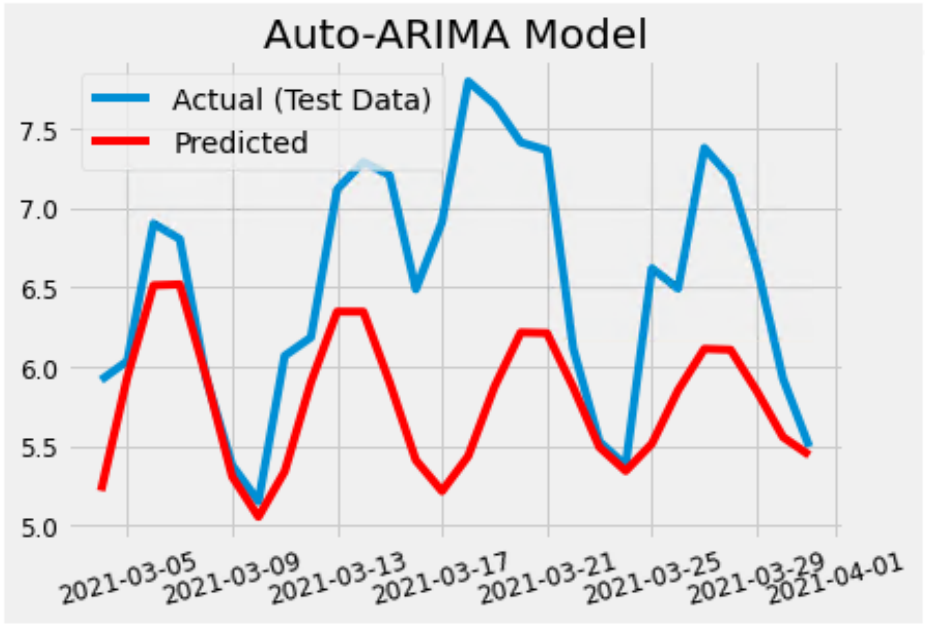



```
evaluate_forecast(test, predictions)
```

| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | 0.283518 | 0.491669 | 0.393444 | 0.400674 | 7.479442 | 0.632988 |

ARIMA Model with TS Log (With Exog Parameters)

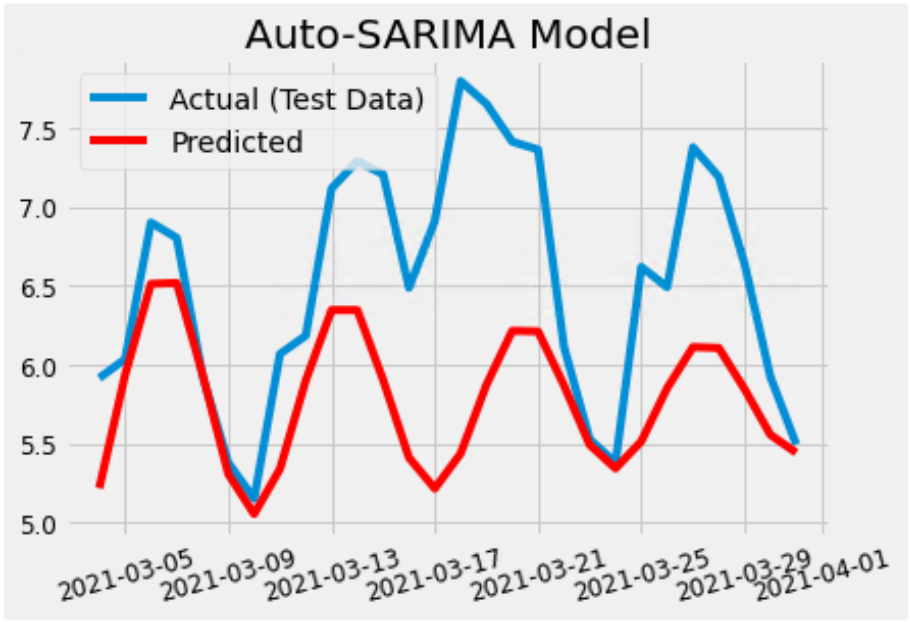
# Modelling Test for Daily Prediction (Auto Arima & Auto Sarima)




```
evaluate_forecast(test, forecast)
```

| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | -0.609361 | 0.732946 | 0.711513 | 0.899993 | 10.50775 | 0.94868 |

Auto Arima Model With TS Diff
(With Exog Parameters)




```
evaluate_forecast(test, forecast)
```

| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | -0.609361 | 0.732946 | 0.711513 | 0.899993 | 10.50775 | 0.94868 |

Auto Sarima Model with TS Log
(With Exog Parameters)

# Modelling Test for Daily Prediction (Tune Sarimax & Tune Sarima)

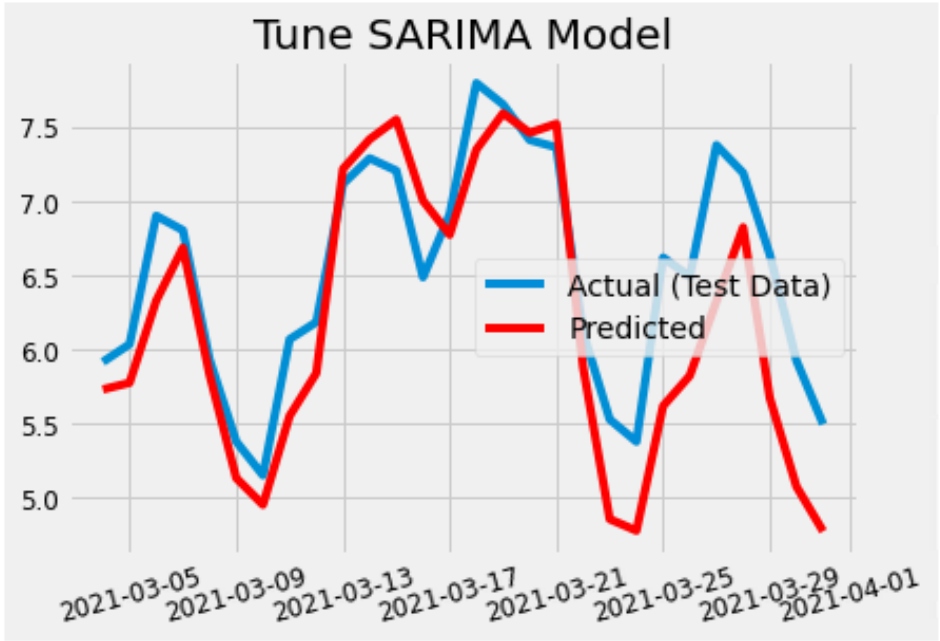
```
p = d = q = range(0,3)
pdq = list(itertools.product(p,d,q))
seasonal_pdq = [(x[0],x[1],x[2],7) for x in list(itertools.product(p,d,q))]
print('Examples of parameter combinations for Seasonal ARIMA...')
print('SARIMAX: {} x {}'.format(pdq[1], seasonal_pdq[1]))
print('SARIMAX: {} x {}'.format(pdq[1], seasonal_pdq[2]))
print('SARIMAX: {} x {}'.format(pdq[2], seasonal_pdq[3]))
print('SARIMAX: {} x {}'.format(pdq[2], seasonal_pdq[4]))
```

```
Examples of parameter combinations for Seasonal ARIMA...
SARIMAX: (0, 0, 1) x (0, 0, 1, 7)
SARIMAX: (0, 0, 1) x (0, 0, 2, 7)
SARIMAX: (0, 0, 2) x (0, 1, 0, 7)
SARIMAX: (0, 0, 2) x (0, 1, 1, 7)
```

```
# Tuning to find the model with least AIC score
min_aic = 999999999
for param in pdq:
    for param_seasonal in seasonal_pdq:
        try:
            mod = sm.tsa.statespace.SARIMAX(train,
                                            order = param,
                                            seasonal_order = param_seasonal,
                                            enforce_stationarity = False,
                                            exog = exo_train, #add-on with exogenous variable
                                            enforce_invertibility = False,freq ='D')


            results = mod.fit(maxitter = 500)
            print('ARIMA{}x{}12 - AIC:{}'.format(param,param_seasonal,results.aic))

            #check for best model with lovest AIC
            if results.aic < min_aic:
                min_aic = results.aic
                min_aic_model = results
        except:
            continue
```




```
evaluate_forecast(test, predictions)
```

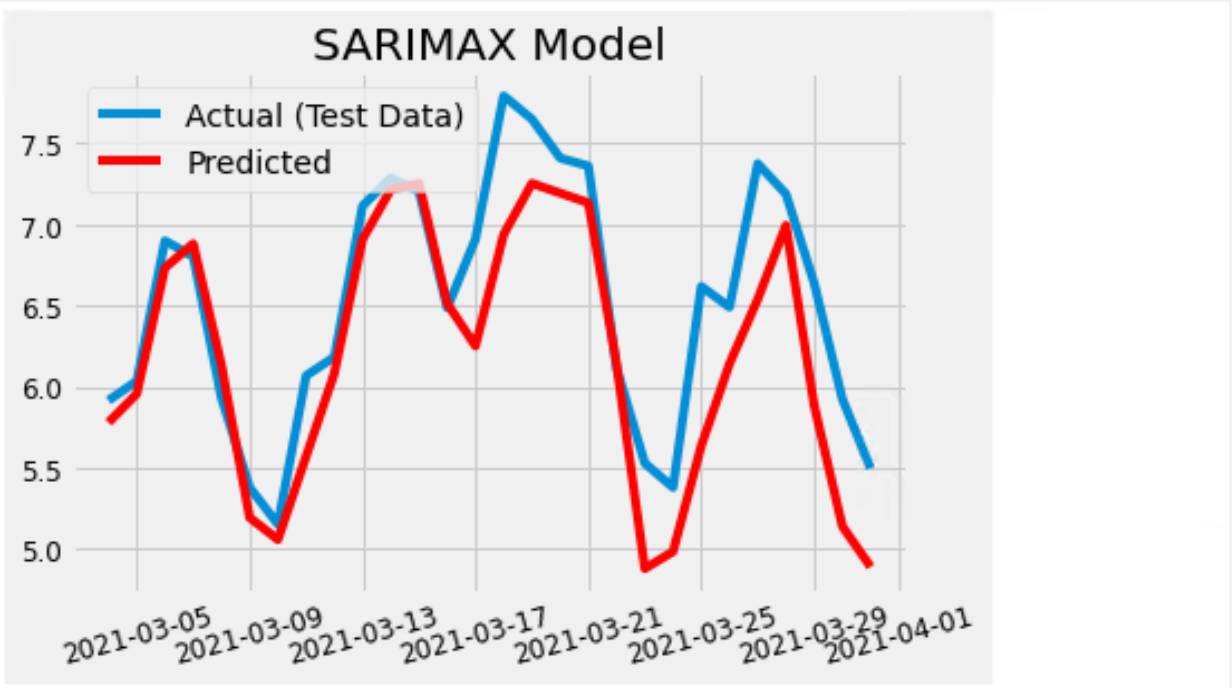
| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | 0.533246 | 0.41424 | 0.343798 | 0.26102 | 6.521452 | 0.510901 |

Example of Tune SARIMAX Model With Ts Log, Different Combination of PDQ, Seasonal PDQ (With Exog Parameters)

Tune Sarima Model with TS Log (With Exog Parameters)

# Modelling Test for Daily Prediction (Sarimax)

Sarimax Model with TS Log(With Exog Parameters)



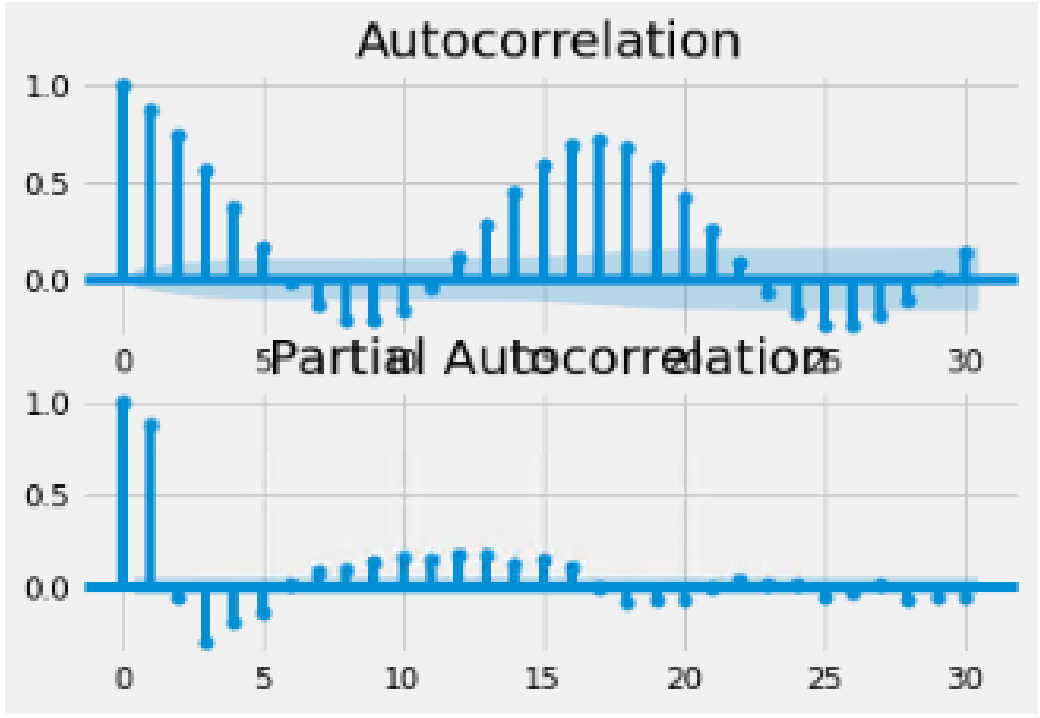

```
evaluate_forecast(test, predictions)
```

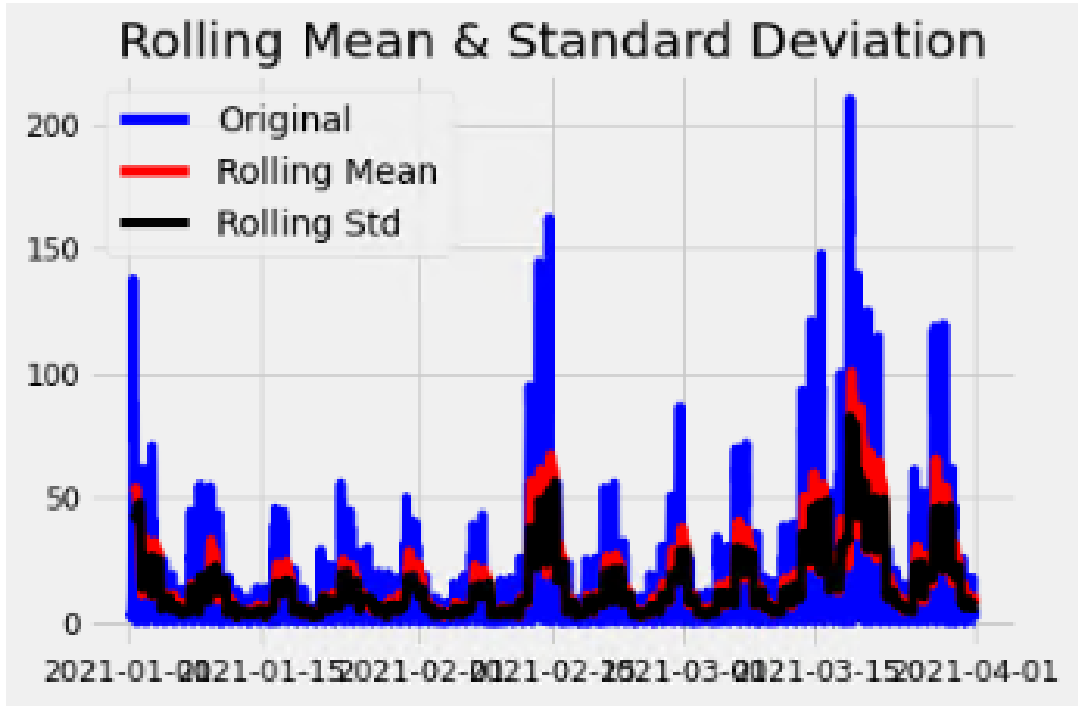
| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | 0.62832 | 0.350706 | 0.210924 | 0.207852 | 5.417608 | 0.455908 |

# **Appendix:** Modelling Test For Hourly Prediction

# Preprocessing for Modelling Test for Hourly Prediction

```
# plot ACF & PACF charts
plt.figure()
plt.subplot(211)
plot_acf(df_ts, ax=plt.gca(),lags=30)
plt.subplot(212)
plot_pacf(df_ts, ax=plt.gca(),lags=30)
plt.show()
```



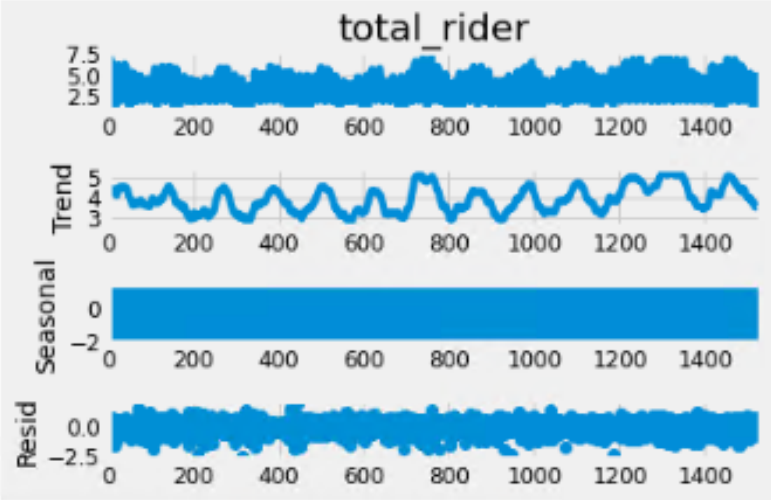




```
Results of Dickey-Fuller Test:
Test Statistic                  -4.087250
p-value                          0.001017
#Lags Used                      19.000000
Number of Observations Used    1510.000000
Critical Value (1%)              -3.434688
Critical Value (5%)              -2.863456
Critical Value (10%)             -2.567790
dtype: float64
```

# Preprocessing for Modelling Test for Hourly Prediction

## Decomposition

```
# Decompose the data frame to get the trend, seasonality and noise
decompose_result = seasonal_decompose(df_ts,model='additive',period=17)
decompose_result.plot()
plt.figure(figsize = (15,6))
plt.show()
```

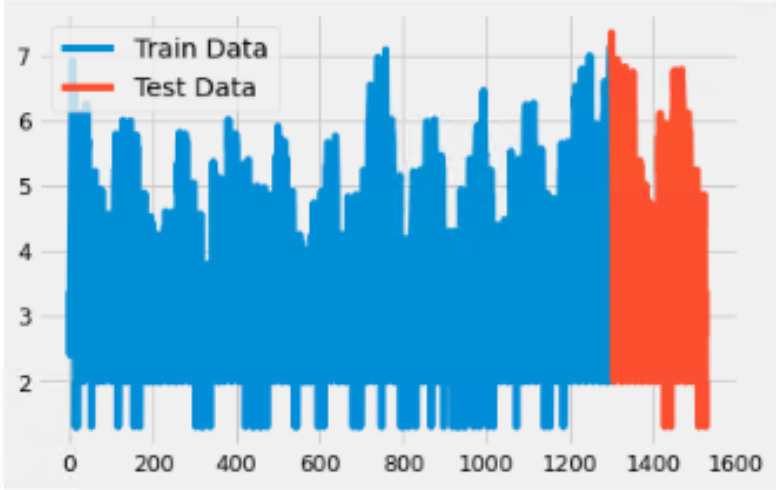



## split dataset

```
## get train & test data
train_size = int(len(df_ts) * 0.85)
train, test = df_ts[0:train_size], df_ts[train_size:]
```

```
#plotting the data
plt.plot(train, label='Train Data')
plt.plot(test, label = 'Test Data')
plt.legend()
```

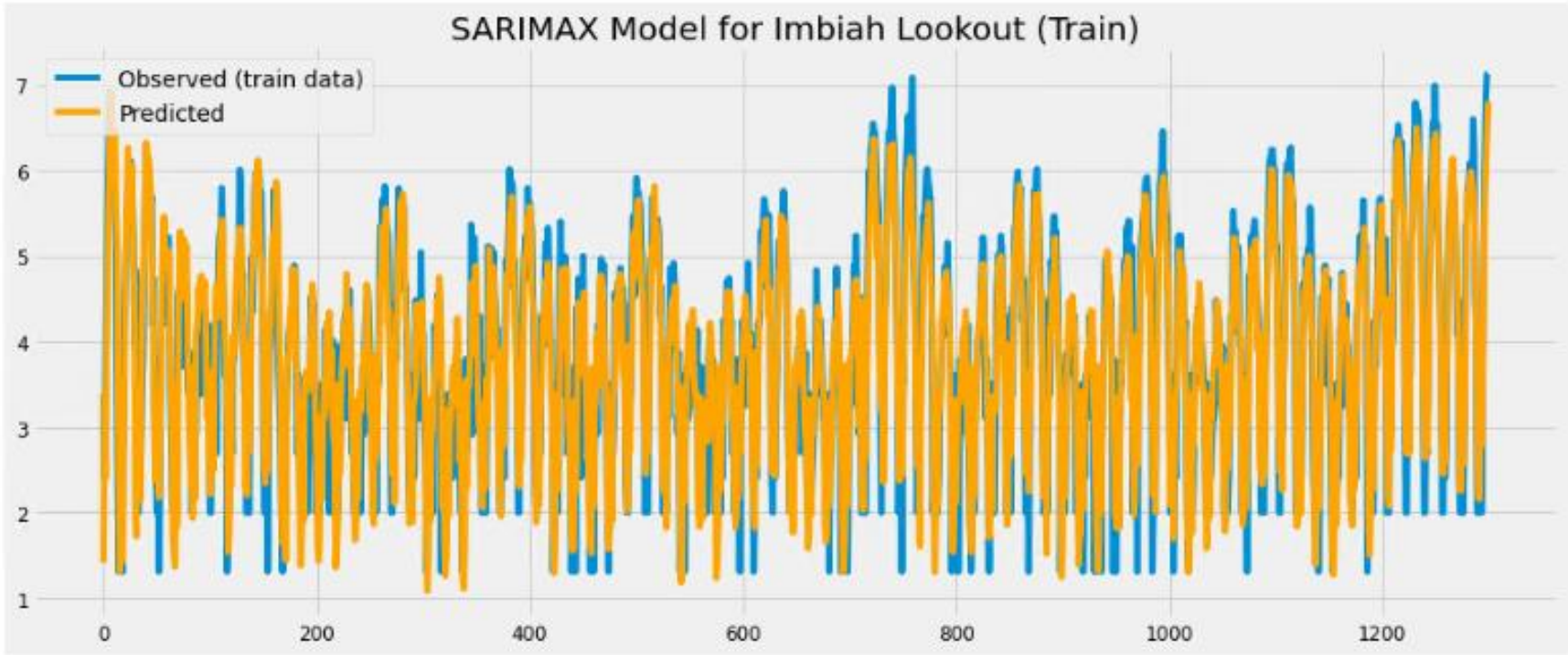
```
<matplotlib.legend.Legend at 0x26d9e82fd48>
```




```
#splitting the exog variable

exog_train_size = int(len(df_exog) * 0.85)
exog_train, exog_test = df_exog[0:exog_train_size], df_exog[exog_train_size:]
```

# Modelling Test for Hourly Prediction – Imbiah Lookout (Sarimax - Train)

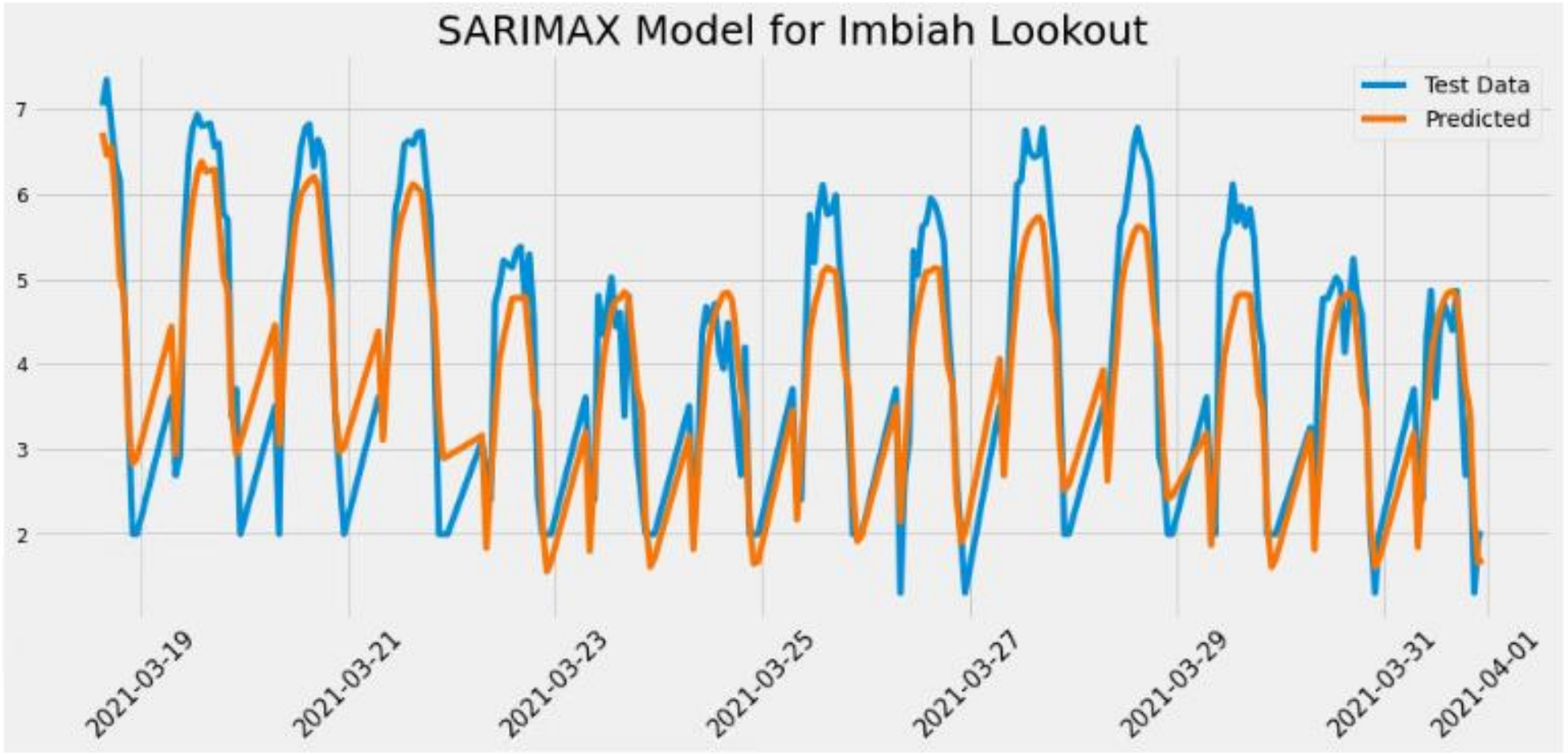



```
# get evaluation metrics
evaluate_forecast(train, sarimax_train_predictions)
```

| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | 0.811222 | 0.456604 | 0.370949 | 0.353466 | 15.097073 | 0.59453 |

```
## visualize actual & predicted values
plt.figure(figsize = (15,6))
plt.plot(test, label='Observed (test data)')
plt.plot(sarimax_predictions, color='orange', label = 'Predicted')
plt.title('SARIMAX Model')
plt.legend()
```

# Modelling Test for Hourly Prediction – Imbiah Lookout (Sarimax - Test)

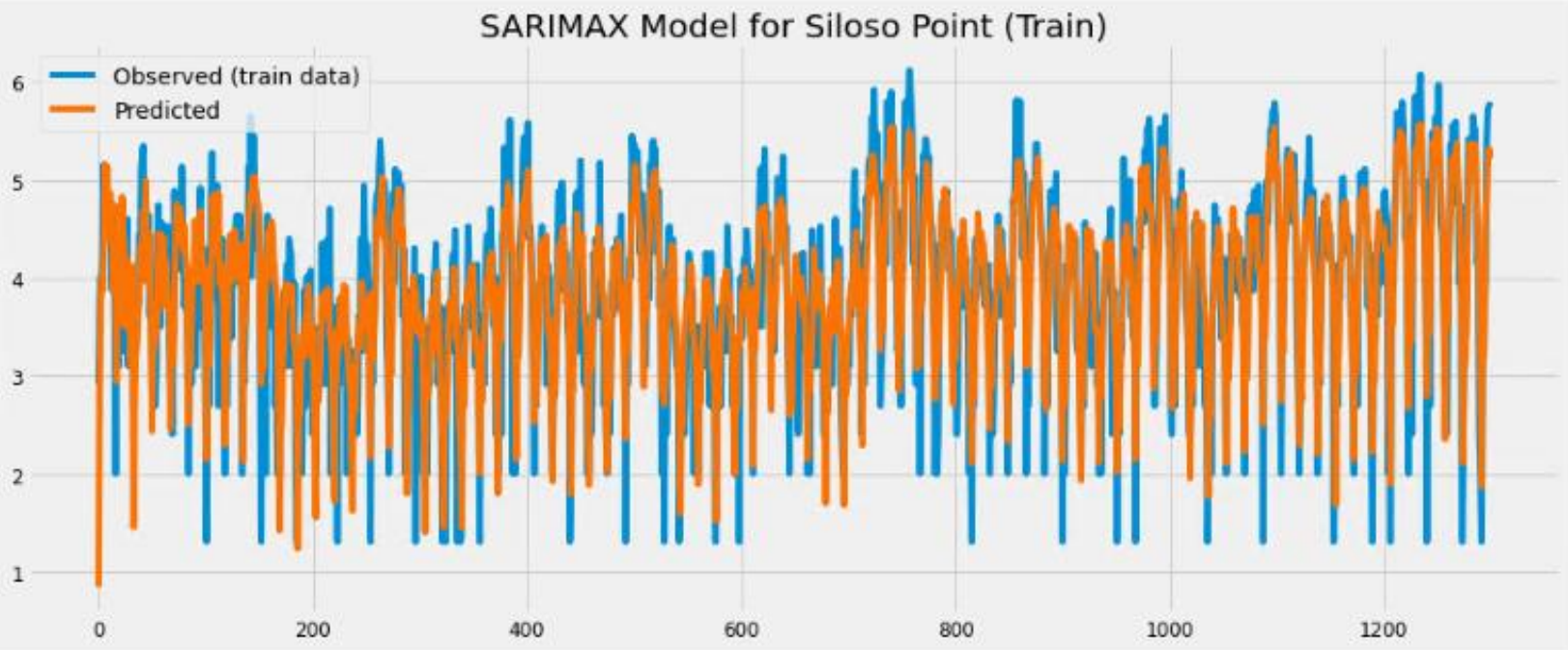



| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | 0.800854 | 0.625101 | 0.608202 | 0.519479 | 16.24869 | 0.720749 |

# Modelling Test for Hourly Prediction – Siloso Point (Sarimax - Train)

```
## visualize actual & predicted values
plt.figure(figsize = (15,6))
plt.plot(train, label='Observed (train data)')
plt.plot(sarimax_train_predictions, color='#F97306', label = 'Predicted')
plt.title('SARIMAX Model for Siloso Point (Train)')
plt.legend()
```

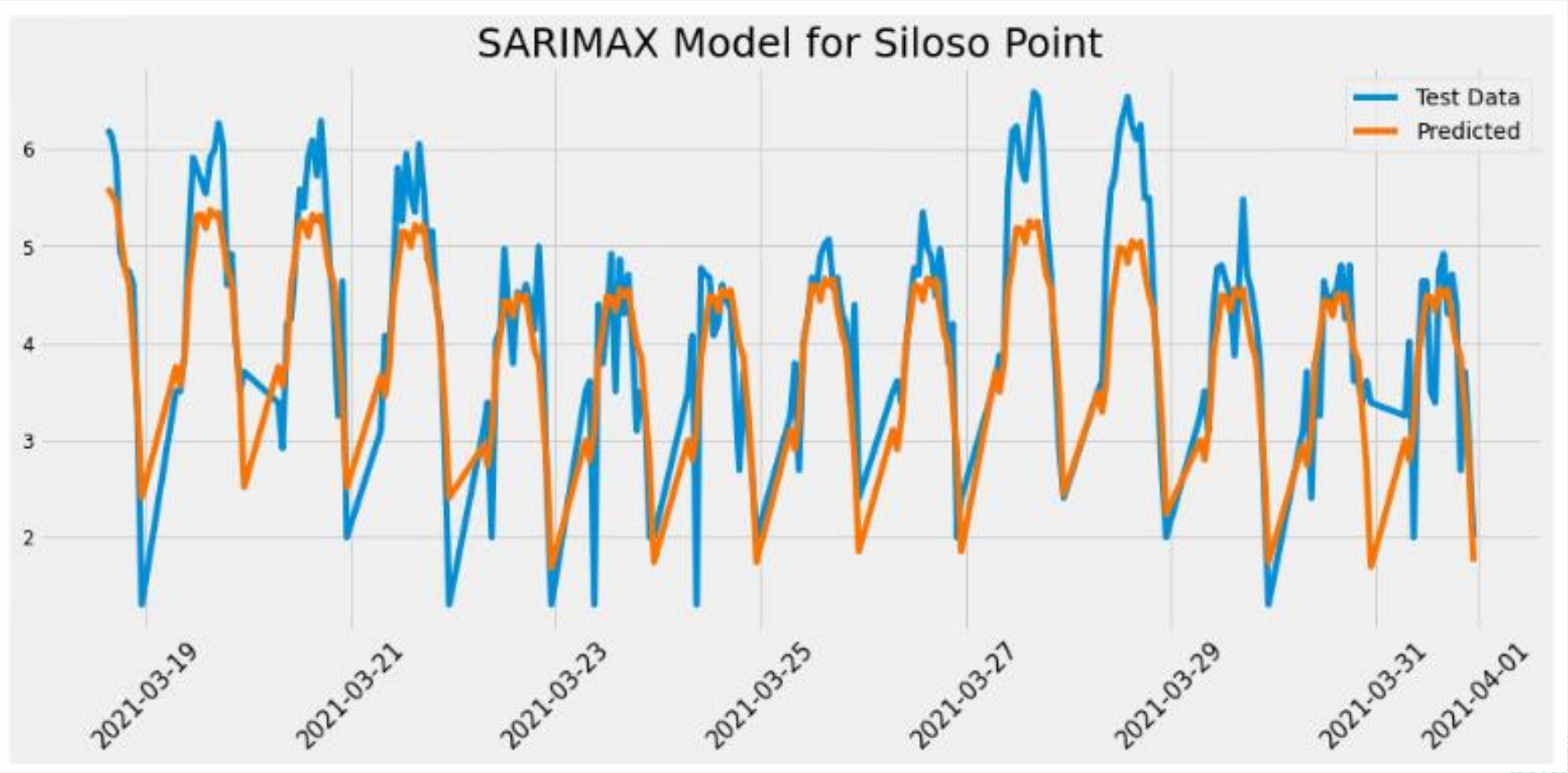
```
<matplotlib.legend.Legend at 0x22007317188>
```




```
# get evaluation metrics
evaluate_forecast(train, sarimax_train_predictions)
```

| | r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|---|
| 0 | 0.666248 | 0.443233 | 0.370646 | 0.32537 | 14.156257 | 0.570412 |

# Modelling Test for Hourly Prediction – Siloso Point (Sarimax - Test)




| r2_score | mean_absolute_error | median_absolute_error | mse | mape | rmse |
|---|---|---|---|---|---|
| 0.7 | 0.5 | 0.39 | 0.41 | 13.49 | 0.64 |


RISE
BY
DIGITAL BCG
ACADEMY