Team Members:

a) Aashi Aashi
b) Sharon Liu
c) Saurabh Arora
d) Anthony Moreno

TIME SERIES FORECASTING – DAILY GROCERY SALES

Description of project:

The purpose of our project is to predict daily sales for stores located in Ecuador by using time series forecasting. The data comes from an Ecuadorian company known as Corporacion Favorita. 6 datasets were used in this project. The training data includes dates, store and product information, whether that item was being promoted, as well as sales numbers. The ***family*** column identifies the type of product sold. The ***sales*** column gives the total sales for a product family at a particular store on a given date. The ***onpromotion*** column shows the total number of items in a product family that were being promoted at a store on a given date. Transaction data has the information for the number of transactions made by a certain store on a given date, which is highly correlated with the sales data in the training data. Holiday and events data contains the type of holiday on a given date. This data is valuable to understand past sales, trends, and seasonality components. The oil data contains the daily oil price. Since Ecuador is an oil-dependent country and its economic health is highly vulnerable to oil price shocks, this data is valuable for understanding how oil prices affect the sales of each product family differently.
We choose a real-world example as the project topic to understand how machine learning is changing the world and making life easier. Forecasting on grocery sales could help ensure retailers improve customer satisfaction and reduce waste by just having enough of the right products at the right time, which could be applied to the real market and positively affect people's real life.

Exploratory Data Analysis

Our main objective is to predict overall daily sales for the grocery company. For this reason, sales data should be examined in great detail; seasonality, trends, anomalies, and similarities of the sales will all be discussed in this section. From the plot (figure 1.1), we can see the total sales for all the years i.e., 2013 to 2017. The year 2016 has the highest sales among all; 2017 has the lowest – one reason could be that the sales data is present only till August 2017

Figure 1.2 shows the sales distribution across different months for all the years to check if the same monthly trend is present across all the years. We could

see that all years except 2016 have a dip in sales in April. One reason for the spike in sales in April 2016 could be the earthquake that happened on April 16th, 2016

Looking at Figure 2.1, Figure 2.2, and Figure 2.3, we can see the "Holiday" type is the most common Holiday followed by the "Event" Type and "Additional". Also, we can see that number of Local and National Holidays are pretty comparable in the dataset

Looking at Figure 3.1, and Figure 3.2 we can see that most numbers of stores fall in Store Type D and C; and in Clusters 3, 6, and 10.

Hypothesis Testing

Looking at the data, we formed 4 hypotheses and performed EDA and Anova Test to see if the Hypothesis can be statistically accepted.

1. *Hypothesis 1*: Store affects sales
   Looking at Fig 4.1, 4.2, and 4.3 we can see that most of the store has the same total sales except Store 44-51. We further investigated these stores and found out that all have a ~25% increase in sales in 2016 and a spike in April.
2. *Hypothesis 2*: Location affects sales
   From Fig 5.1, 5.2, and 5.3, we can see that some cities have much higher average total sales than others
3. *Hypothesis 3*: Holiday affects sales
   Looking at Fig 6.1, 6.2, and 6.3, even though total sales on days without holiday is far greater than on days with the holiday; if we investigate average sales; it is just the opposite.
   Conclusion:
   Sales are much higher on Holidays than without Holidays
4. *Hypothesis 4*: Weekday affects sales
   Looking at Fig 7.1, 7.2, and 7.3, Total sales on weekends are far greater than that on weekdays; if we investigate average sales the same trend continues.

Preprocessing Time Series for Forecasting

As seen in Fig 8.1, and 8.2, we can see an increasing trend along with a spike in sales during a particular period representing seasonality. To check if our data is stationary or not, we performed ADF Test. Fig 8.3 contains the result of the test, ad since the p value>0.05 we failed to reject the Null Hypothesis; hence data is not stationary. To see the presence of trend and seasonality and

trend in the dataset, we decomposed the data into seasonality, trend, and residuals (Fig 8.4).

Since Time series forecasting can only be done on stationary data, we need to convert the same. For the same, we performed first-degree differencing and checked again for stationarity using ADF (Fig 8.5 and Fig 8.6). In the ADF test, our p_value is coming to less than 0.05; statistically proving that the time series is stationary now. Since first-degree differencing made our series stationary, the d value in the ARIMA model would be 1.

To find Auto-Regressive(p) and Moving Average(q) for ARIMA, we plotted PACF and ACF on the first-degree differencing time series. From the plots, we took p and q as 2.

Modeling - Time Series Forecasting

MODEL 1 -*ARIMA*:

After finding the components of ARIMA i.e. p,d,q values, we ran the ARIMA model on our dataset with the following parameters:

- AR (p=2), the number of lag observations or autoregressive terms in the model
- Integration (d)=1, the difference in the nonseasonal observations
- MA (q)=2, the size of the moving average window.

A summary of the fitted model is shown in Fig 9.1. From Fig 9.2 and 9.3, the residual errors seem fine with near zero mean and uniform variance. Fig 9.4 shows the actual and forecasted sales; looking at the graph, ARIMA works pretty well.

To find the accuracy of our forecasted data, we calculated MAPE (Mean Absolute Percentage Error) (Fig 9.5), and it's coming to 0.264 i.e., we are getting 74% accuracy for the forecasted values

MODEL 2 -*AUTO ARIMA*:

We ran AUTO-ARIMA on our time series. One major difference between ARIMA and AUTO ARIMA is that it automatically calculated p,d, and q values by finding the permutation with the lowest AIC score. Also, it accounts for the seasonality aspect while fitting the data set into the model.

Looking at Fig 10.1 and 10.2, we can see that AUTO ARIMA automatically finds p,d,q value; in our case p=3,d=0,q=2. Fig 10.3 shows the forecasted sales from AUTO ARIMA and actual sales

MODEL 3 *-XGBRegressor with Grid Search Cross Validation*:

We finally tested our model on XGBRegressor with 4-fold cross validation and found the optimum parameters as max_depth = 4 and n_estimators as 500.

With these parameters we are achieving the Mean Average Error as 154.34 and Root Mean Squared Log Error as 0.1. These values mean that XGBRegressor provides promising results.

Solution and Insights:
Seeing as our problem was a time-series problem that required forecasting, our solution would logically be a regression. In this case, we decided to use ARIMA and AUTO-ARIMA to predict sales.

Appendix

Figure 1.1

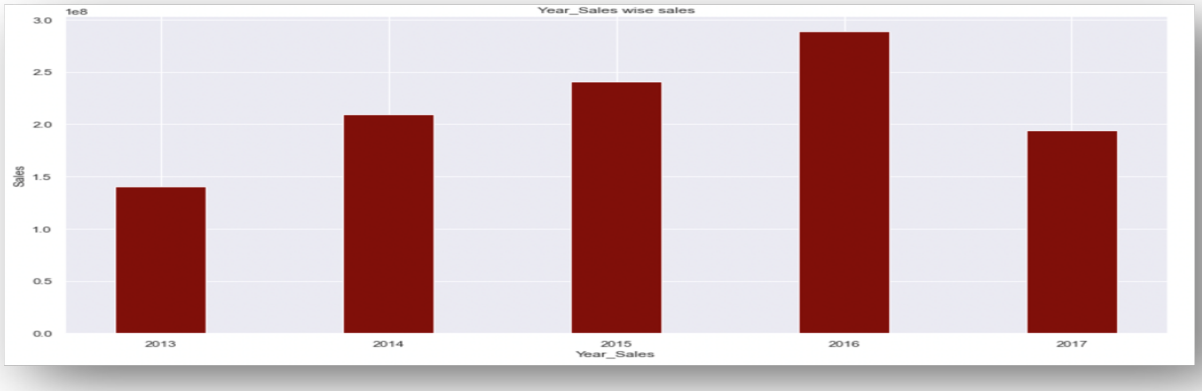


Figure 1.2

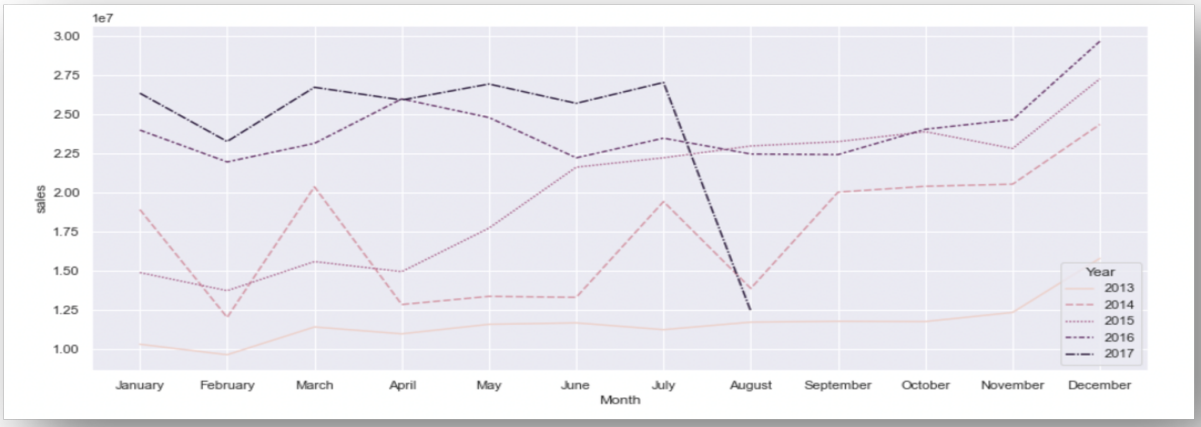


Figure 1.3

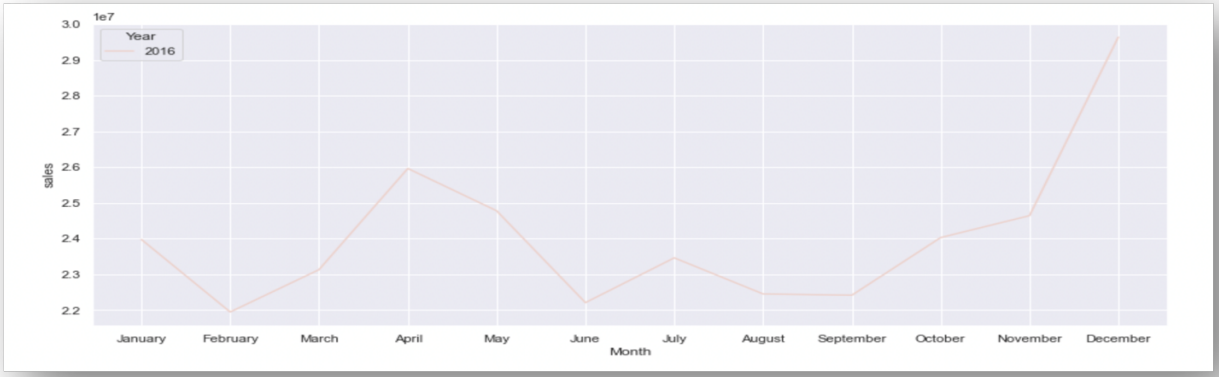


Figure 2.1

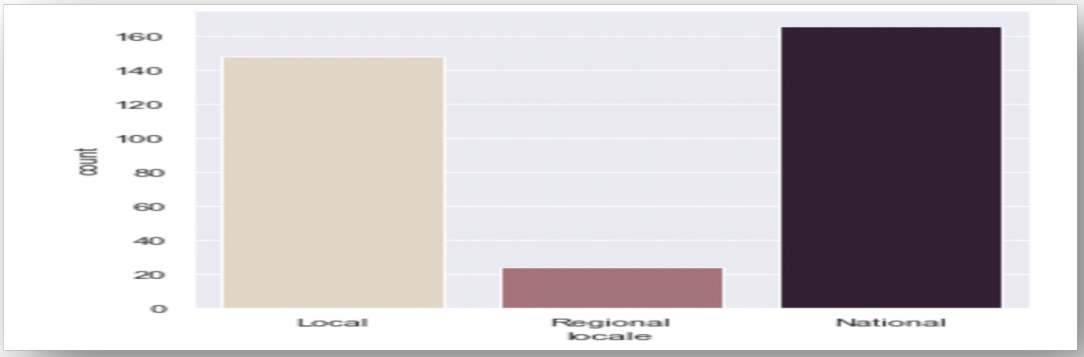


Figure 2.2

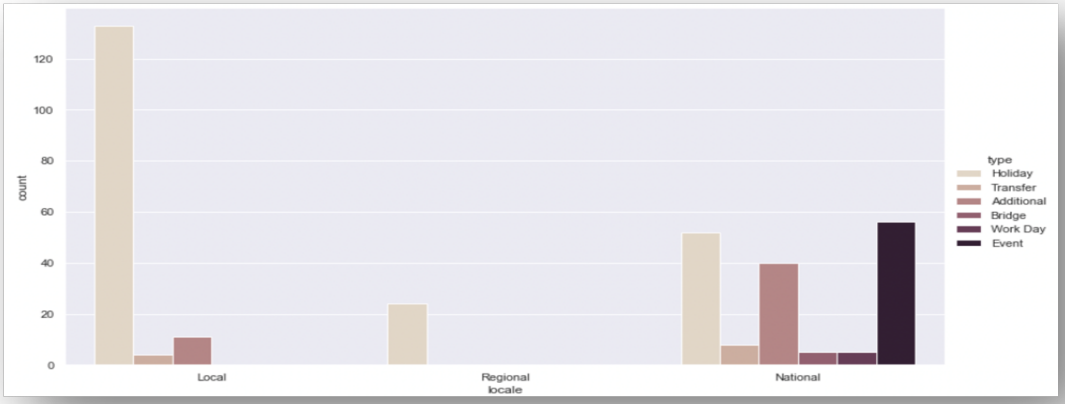


Figure 2.3

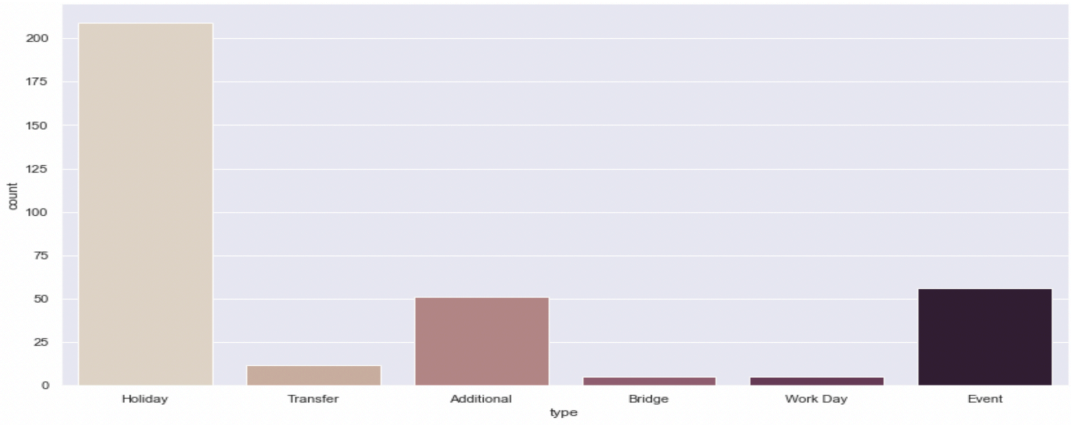


Figure 3.1

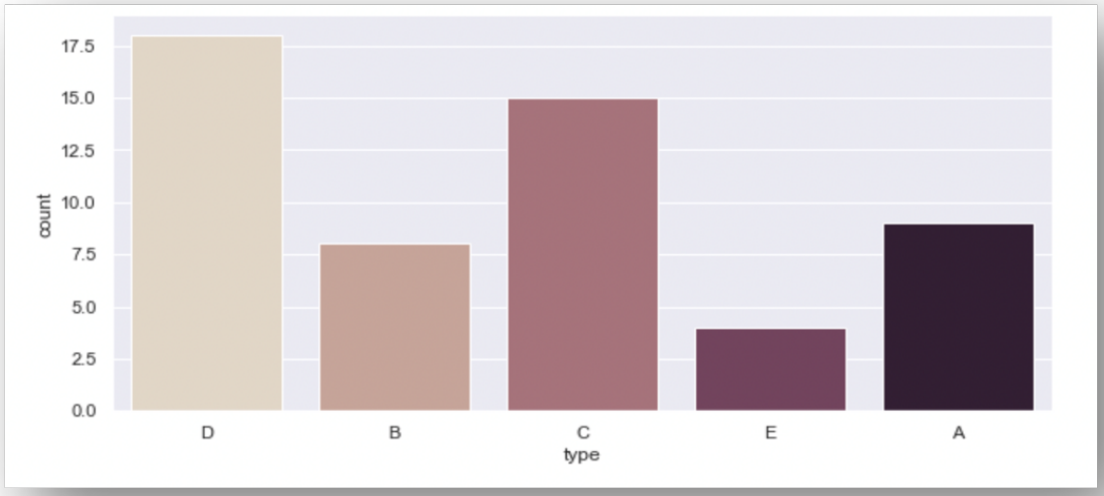


Figure 3.2

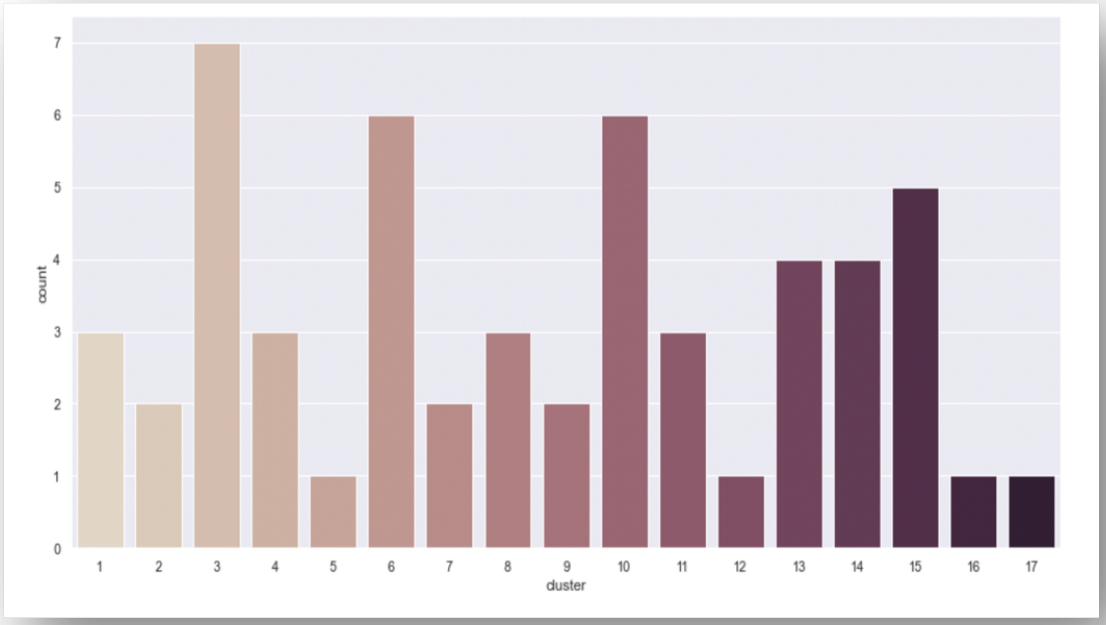


Figure 4.1

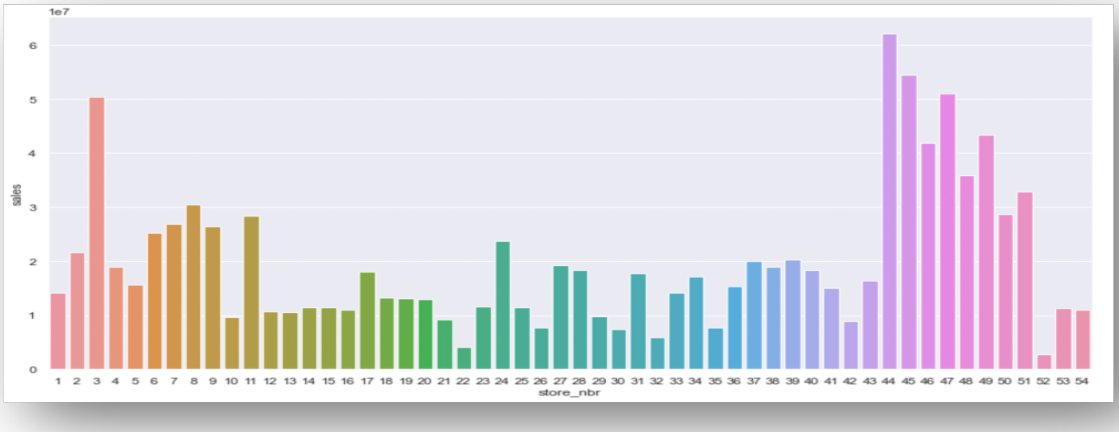


Figure 4.2

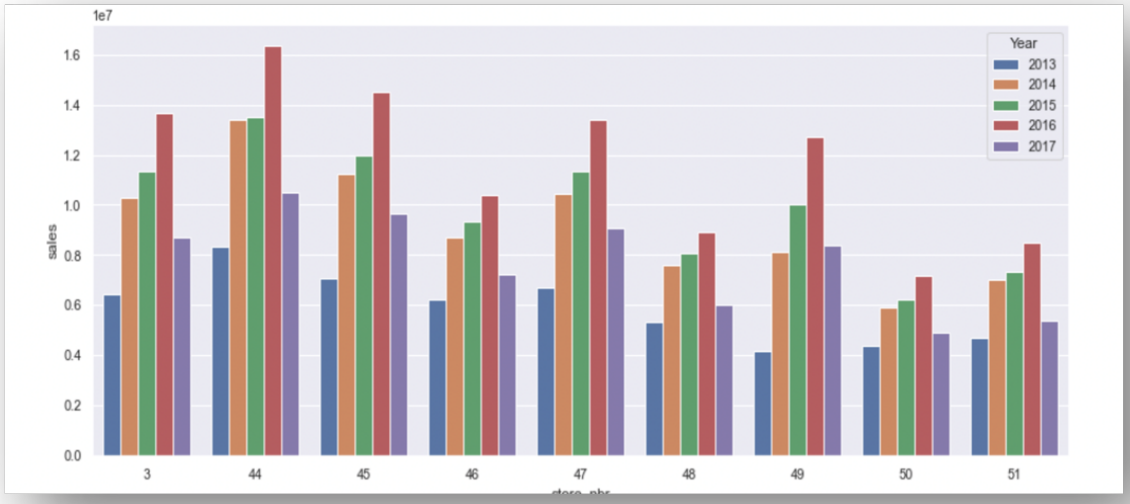


Figure 4.3

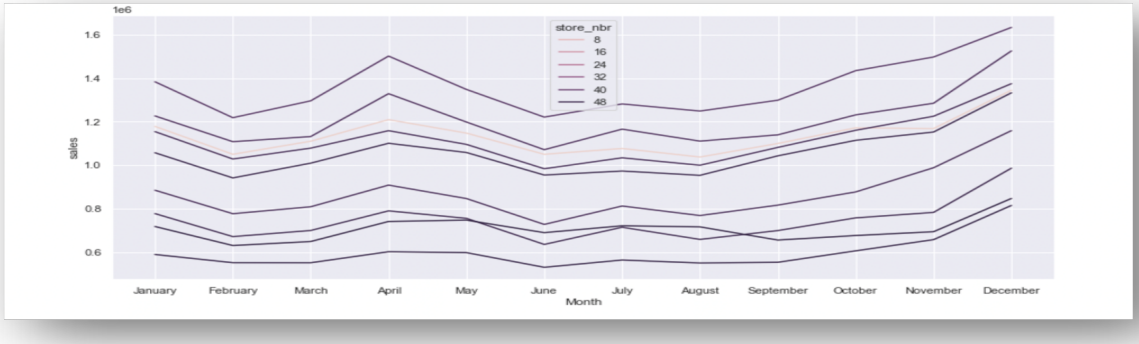


Figure 4.4

```
# Ordinary Least Squares (OLS) model
model = ols('sales ~ store_nbr', data=train_df4).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
anova_table
```

	sum_sq	df	F	PR(>F)
store_nbr	1.685856e+11	53.0	2746.286055	0.0
Residual	3.475688e+12	3000834.0	NaN	NaN

Figure 5.1

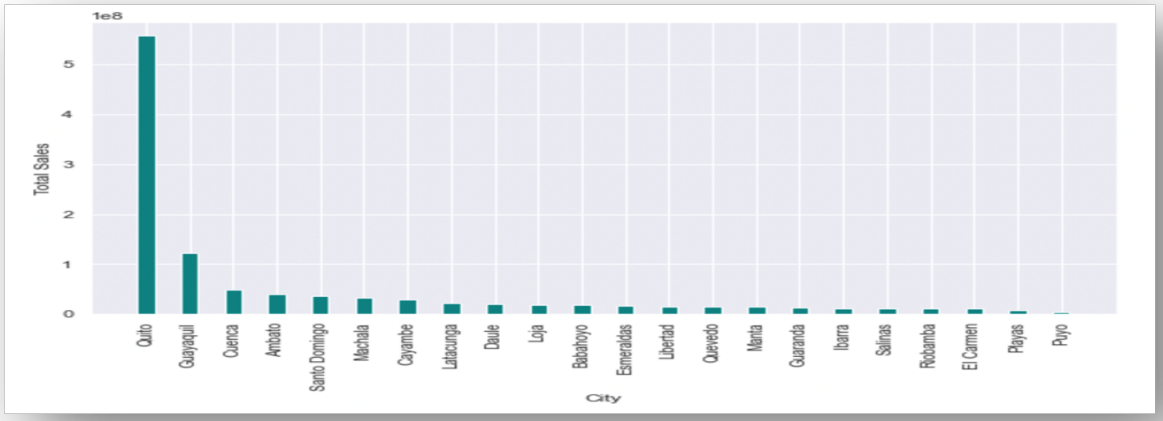


Figure 5.2

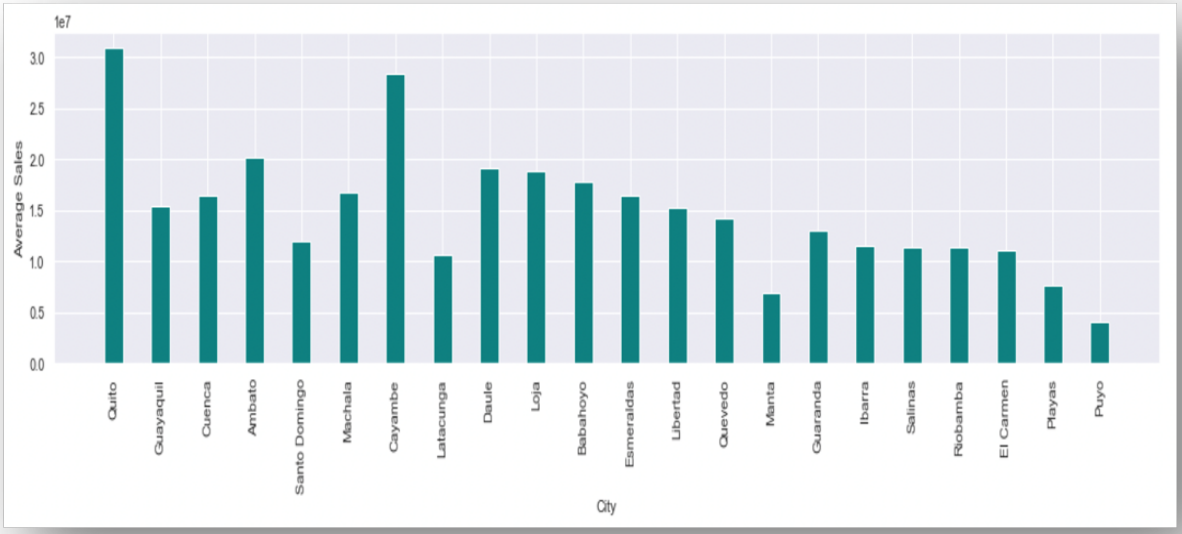


Figure 5.3

```
# Ordinary Least Squares (OLS) model
model = ols('sales ~ state', data=train_df4).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
anova_table
```

	sum_sq	df	F	PR(>F)
state	7.012482e+10	15.0	3925.141397	0.0
Residual	3.574149e+12	3000872.0	NaN	NaN

```
# Ordinary Least Squares (OLS) model
model = ols('sales ~ city', data=train_df4).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
anova_table
```

	sum_sq	df	F	PR(>F)
city	7.185915e+10	21.0	2874.401888	0.0
Residual	3.572415e+12	3000866.0	NaN	NaN

Figure 6.1

Figure 6.2

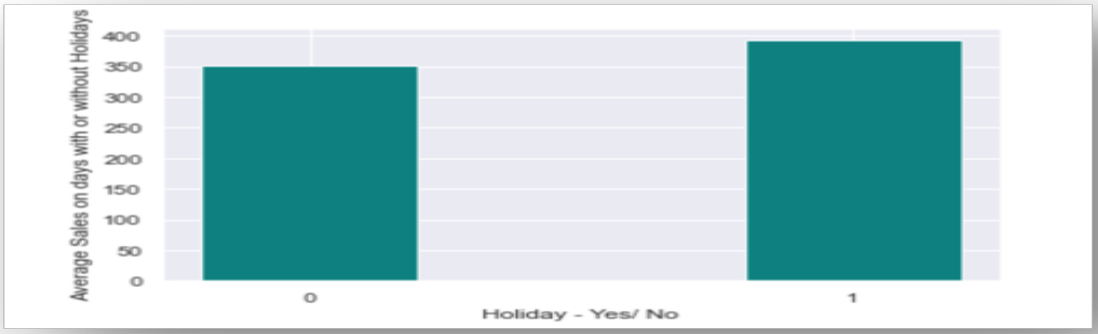


Figure 6.3

```
# Ordinary Least Squares (OLS) model
model = ols('sales ~ holiday_type', data=train_df4).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
anova_table
```

	sum_sq	df	F	PR(>F)
holiday_type	6.132023e+08	1.0	505.027878	7.821313e-112
Residual	3.643661e+12	3000886.0	NaN	NaN

Figure 7.1

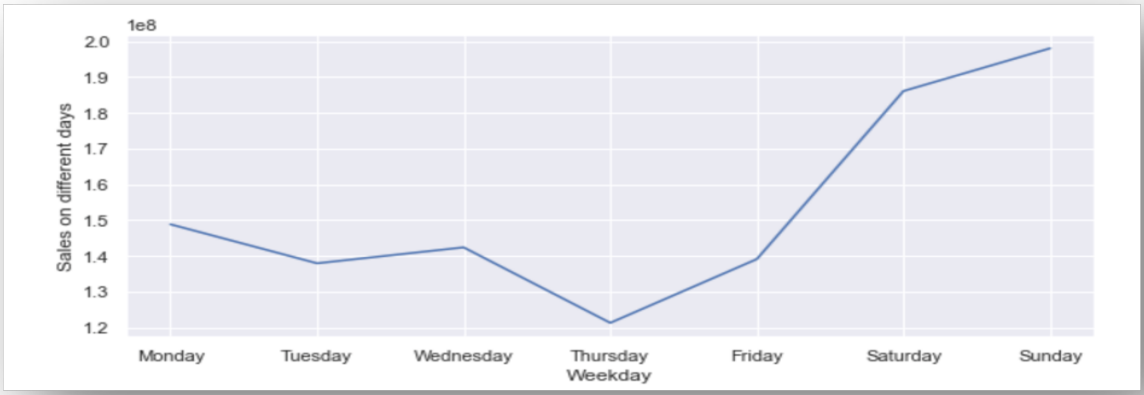


Figure 7.2

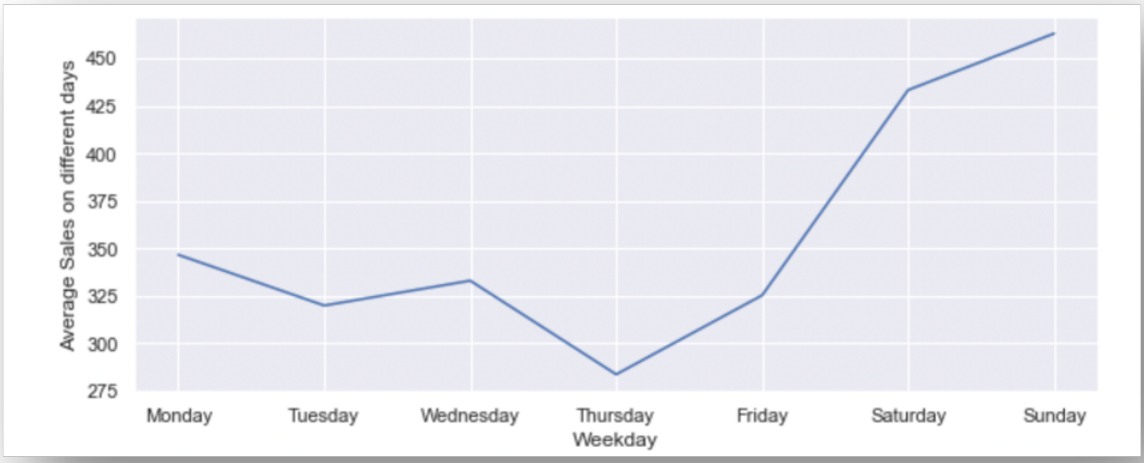


Figure 7.3

```
# Ordinary Least Squares (OLS) model
model = ols('sales ~ weekday', data=train_df4).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
anova_table
```

	sum_sq	df	F	PR(>F)
weekday	1.094443e+10	6.0	1506.558564	0.0
Residual	3.633330e+12	3000881.0	NaN	NaN

Figure 8.1

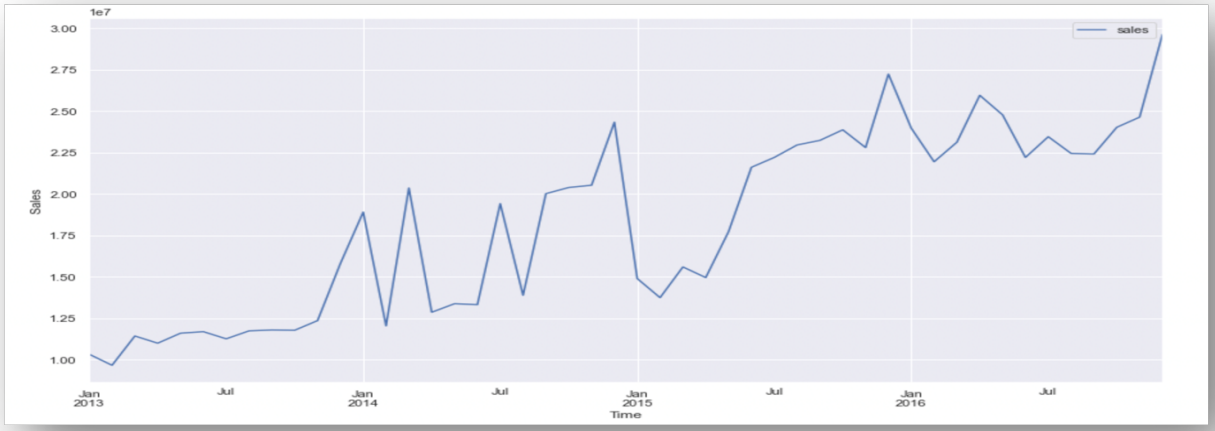


Figure 8.2

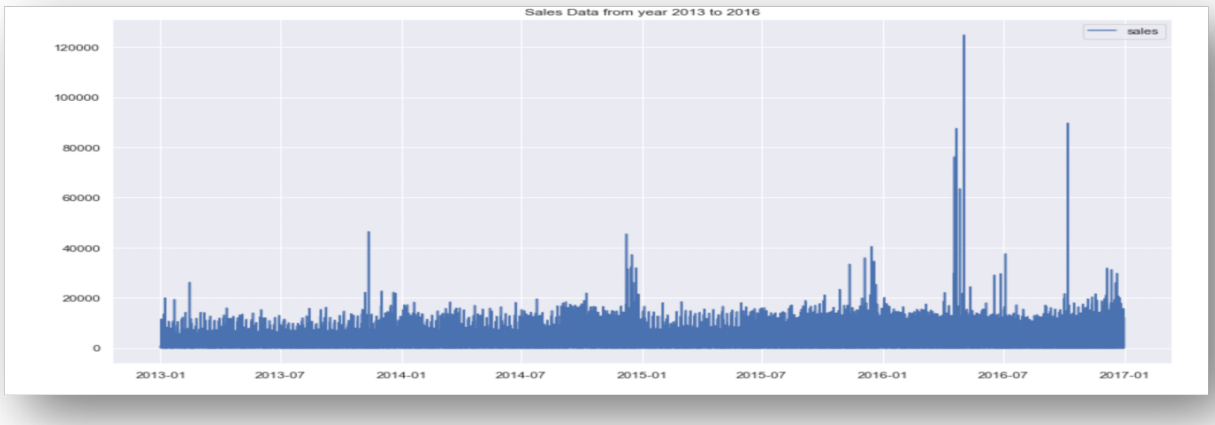


Figure 8.3

```
# ADF Test
# Function to print out results in customised manner
from statsmodels.tsa.stattools import adfuller
dftest = adfuller(overall_dailysales.sales, autolag = 'AIC')
for key, val in dftest[4].items():
    print("\t",key, ": ", val)
```

```
1. ADF :  -2.009672040278186
2. P-Value :  0.2823484153973075
3. Num Of Lags :  22
4. Num Of Observations Used For ADF Regression and Critical Values Calculation : 1434
5. Critical Values :
         1% :  -3.434918371231736
         5% :  -2.8635576234668982
         10% :  -2.5678441693558898
```

Observation: Since p_value>0.5 we can not reject the Null Hypothesis; time series is not stationary

Figure 8.4

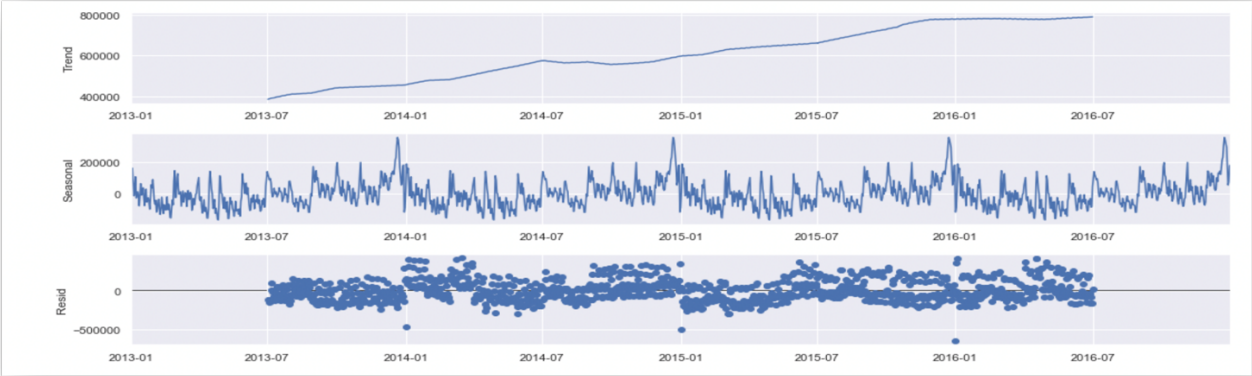


Figure 8.5

```
overall_dailysales['sales_diff'] = overall_dailysales['sales'] - overall_dailysales['sales'].shift(1)
overall_dailysales['sales_diff'].fillna(0, inplace=True)
overall_dailysales[['sales','sales_diff']].head(5)
```

	sales	sales_diff
2013-01-01	2511.618999	0.000000
2013-01-02	496092.417944	493580.798945
2013-01-03	361461.231124	-134631.186820
2013-01-04	354459.677093	-7001.554031
2013-01-05	477350.121229	122890.444136

Figure 8.6

```
# ADF Test
# Function to print out results in customised manner
from statsmodels.tsa.stattools import adfuller
dftest = adfuller(overall_dailysales['sales_diff'], autolag = 'AIC')
for key, val in dftest[4].items():
    print("\t",key, ": ", val)
```

```
1. ADF :  -9.5770669845666
2. P-Value :  2.2088048426585958e-16
3. Num Of Lags :  24
4. Num Of Observations Used For ADF Regression and Critical Values Calculation : 1432
5. Critical Values :
         1% :  -3.4349247631306237
         5% :  -2.8635604442944658
         10% :  -2.5678456715029183
```

As seen, after 1st level of differencing, our data is becoming stationary; hence d=1

Figure 9.1

```
from statsmodels.tsa.arima.model import ARIMA
# 1,1,2 ARIMA Model
model = ARIMA(overall_dailysales['sales'],order=(2,1,2))
model_fit = model.fit()
print(model_fit.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                  sales   No. Observations:                 1457
Model:                 ARIMA(2, 1, 2)   Log Likelihood              -19124.172
Date:                Sun, 07 Aug 2022   AIC                          38258.345
Time:                        17:29:38   BIC                          38284.762
Sample:                             0   HQIC                         38268.201
                                - 1457
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.3104      0.047      6.587      0.000       0.218       0.403
ar.L2         -0.2283      0.034     -6.628      0.000      -0.296      -0.161
ma.L1         -0.5909      0.048    -12.299      0.000      -0.685      -0.497
ma.L2         -0.2974      0.045     -6.600      0.000      -0.386      -0.209
sigma2      1.255e+10   1.13e-12   1.11e+22      0.000    1.25e+10    1.25e+10
===================================================================================
Ljung-Box (L1) (Q):                   0.09   Jarque-Bera (JB):              1369.46
Prob(Q):                              0.77   Prob(JB):                         0.00
Heteroskedasticity (H):               2.52   Skew:                             0.13
Prob(H) (two-sided):                  0.00   Kurtosis:                         7.74
===================================================================================
```

Figure 9.2

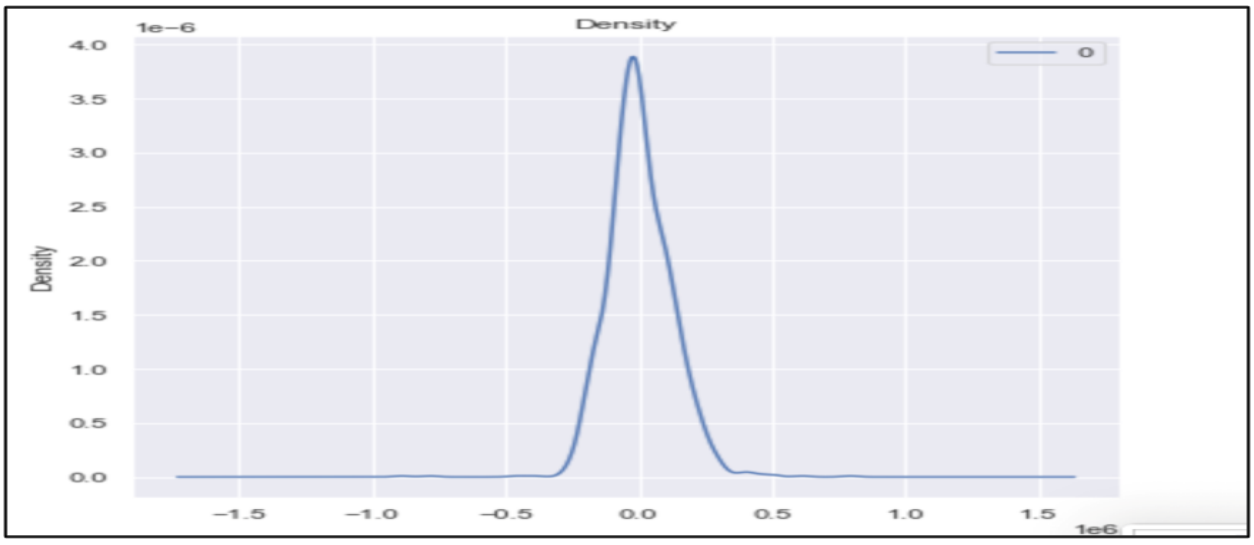


Figure 9.3

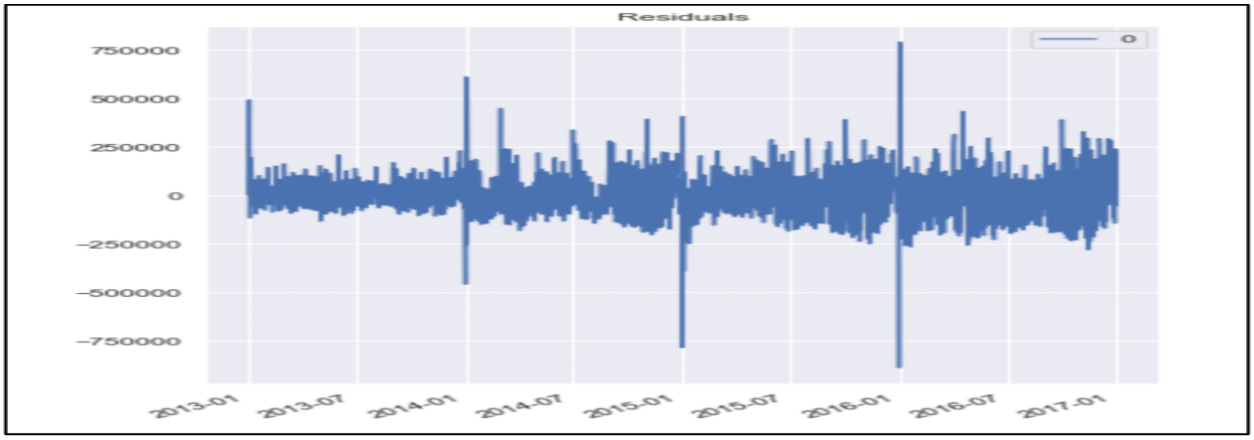


Figure 9.4

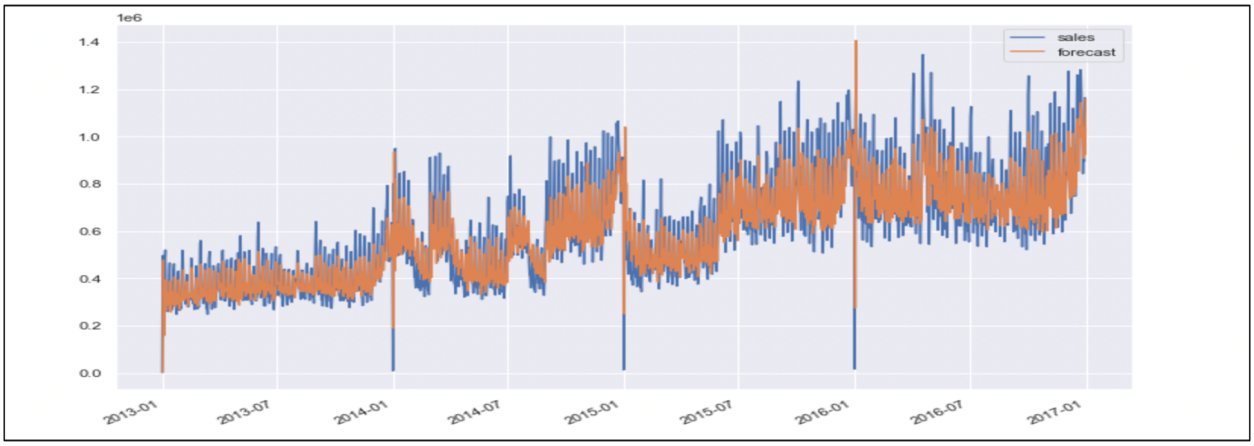


Figure 9.5

```
{'mape': 0.2641178849057131,
 'me': -4426.742652704442,
 'mae': 90965.83439581658,
 'mpe': 0.1360243020661210 4,
 'rmse': 121791.6291369397,
 'corr': 0.8387254368706629,
 'minmax': 0.1345381830209 1303}
```

Figure 10.1

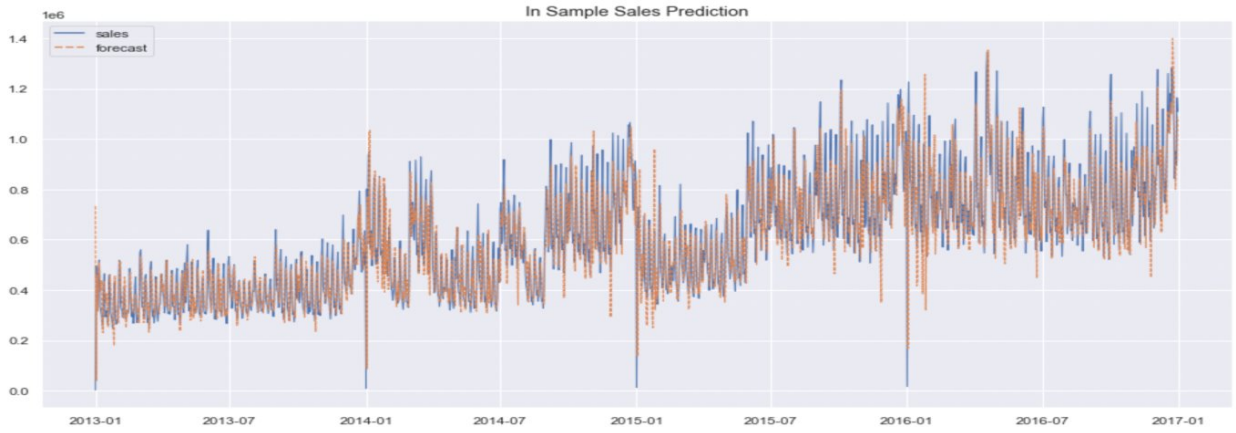
```
model_fit = pm.auto_arima(overall_dailysales['sales'], test='adf',
                          max_p=3, max_d=3, max_q=3,
                          seasonal=True, m=12,
                          max_P=3, max_D=2, max_Q=3,
                          trace=True,
                          error_action='ignore',
                          suppress_warnings=True,
                          stepwise=True)

# summarize the model characteristics
print(model_fit.summary())
```

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[12] intercept   : AIC=38391.738, Time=3.35 sec
 ARIMA(0,0,0)(0,0,0)[12] intercept   : AIC=40028.849, Time=0.02 sec
 ARIMA(1,0,0)(1,0,0)[12] intercept   : AIC=38561.395, Time=0.56 sec
 ARIMA(0,0,1)(0,0,1)[12] intercept   : AIC=39150.765, Time=0.31 sec
 ARIMA(0,0,0)(0,0,0)[12]             : AIC=43110.964, Time=0.01 sec
 ARIMA(2,0,2)(0,0,1)[12] intercept   : AIC=38405.041, Time=0.64 sec
 ARIMA(2,0,2)(1,0,0)[12] intercept   : AIC=38393.145, Time=1.41 sec
 ARIMA(2,0,2)(2,0,1)[12] intercept   : AIC=38430.924, Time=8.40 sec
 ARIMA(2,0,2)(1,0,2)[12] intercept   : AIC=38489.634, Time=9.44 sec
 ARIMA(2,0,2)(0,0,0)[12] intercept   : AIC=38450.225, Time=0.21 sec
 ARIMA(2,0,2)(0,0,2)[12] intercept   : AIC=38402.440, Time=2.21 sec
 ARIMA(2,0,2)(2,0,0)[12] intercept   : AIC=inf, Time=6.30 sec
 ARIMA(2,0,2)(2,0,2)[12] intercept   : AIC=inf, Time=9.51 sec
```

Figure 10.2

```
Best model:  ARIMA(3,0,2)(2,0,1)[12] intercept
Total fit time: 230.597 seconds
                                     SARIMAX Results
==========================================================================================
Dep. Variable:                                   y   No. Observations:                 1457
Model:             SARIMAX(3, 0, 2)x(2, 0, [1], 12)   Log Likelihood              -19045.736
Date:                              Sun, 07 Aug 2022   AIC                          38111.472
Time:                                      17:35:28   BIC                          38164.313
Sample:                                           0   HQIC                         38131.186
                                                - 1457
Covariance Type:                                opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept   6.156e+04   3.33e+04      1.849      0.064   -3698.928    1.27e+05
ar.L1          0.5410      0.012     46.429      0.000       0.518       0.564
ar.L2         -0.5454      0.009    -62.568      0.000      -0.562      -0.528
ar.L3          0.9559      0.013     75.803      0.000       0.931       0.981
ma.L1          0.3319      0.012     26.802      0.000       0.308       0.356
ma.L2          0.8815      0.015     60.309      0.000       0.853       0.910
ar.S.L12      -0.2672      0.056     -4.753      0.000      -0.377      -0.157
ar.S.L24      -0.4669      0.018    -25.881      0.000      -0.502      -0.432
ma.S.L12       0.1931      0.063      3.067      0.002       0.070       0.316
sigma2      1.541e+10      1.925   8.01e+09      0.000    1.54e+10    1.54e+10
==============================================================================
Ljung-Box (L1) (Q):                  18.01   Jarque-Bera (JB):              5966.29
Prob(Q):                              0.00   Prob(JB):                         0.00
Heteroskedasticity (H):               2.48   Skew:                            -0.03
Prob(H) (two-sided):                  0.00   Kurtosis:                        12.91
==============================================================================
```

Figure 10.3


In Sample Sales Prediction
1e6
sales
forecast
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
2013-01
2013-07
2014-01
2014-07
2015-01
2015-07
2016-01
2016-07
2017-01