Supply Chain ARIMA

Luis Parra
ChecoParraLuis@gmail.com

***Abstract*—** The objective of this project is to highlight the comprehensive data science process and develop an ARIMA model for forecasting sales within the context of a supply chain dataset. The report outlines the data cleaning process, exploratory data analysis (EDA), the creation of the ARIMA model, and an analysis of various metrics to evaluate its performance.

1 INTRODUCTION

This dataset was pulled from Kaggle.com. There are 53 variable and over 180,000 observations. The 'order date (DateOrders)' is chosen as the predictor variable and the 'Sales' as the target variable.

2 DATA CLEANING

Despite a minimal data cleaning effort due to the focused variable selection, standard procedures such as handling missing values and duplicates were applied. While specific columns for the model showed no issues, this step reinforces the importance of data cleanliness.

3 EXPLORATORY DATA ANALYSIS

EDA provided valuable insights, particularly in customer segmentation and delivery performance analysis. A pie chart revealed a startling 54% of products experiencing delayed delivery. This can have high financial implications.

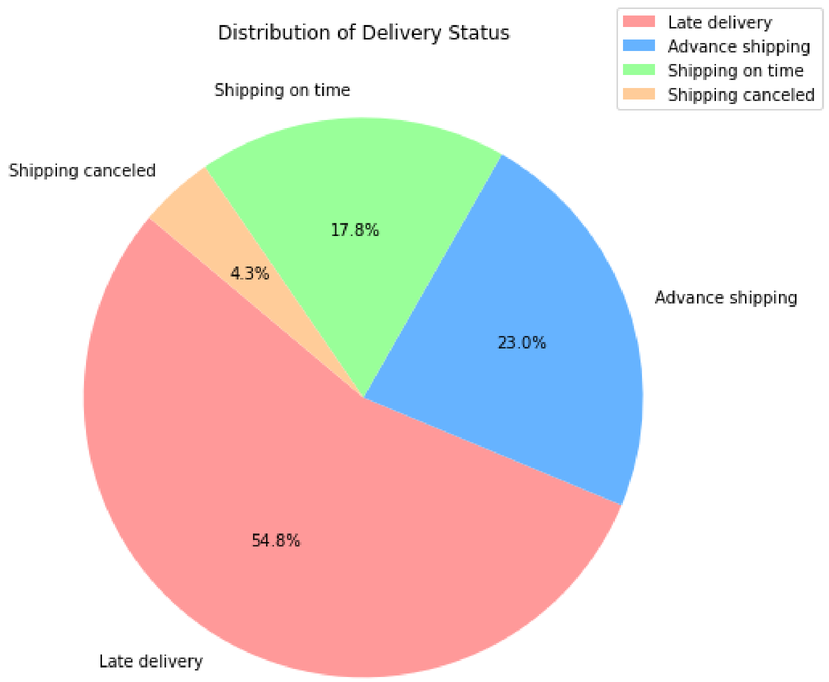


Figure 3.1 Distribution of Deliver Status

Further exploration within the top 5 states confirmed consistency, raising concerns that stakeholders should address this issue.

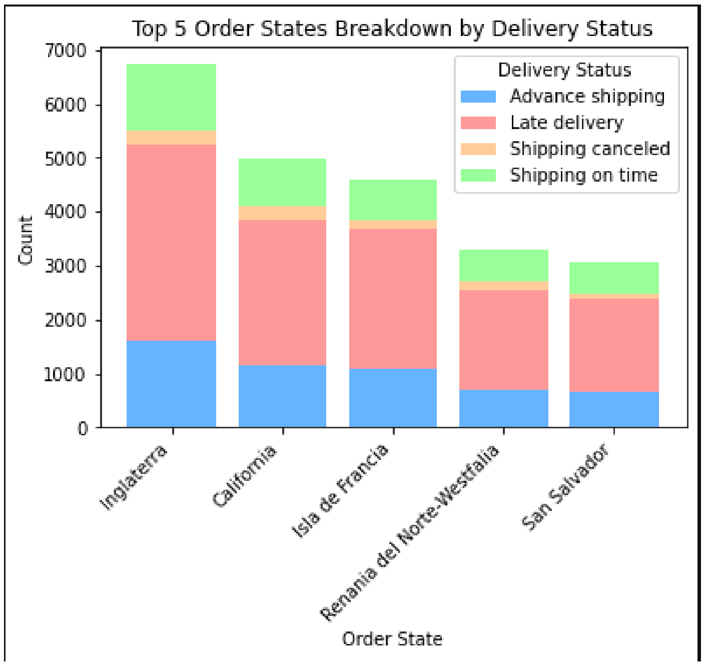


Figure 3.2 Bar Chart of Top 5 States

4 MODEL CREATION

An ARIMA model is created to forecast a specific time period. The model's parameters, denoted as (p, d, q), represent Autoregression (AR), Integration (I), and Moving Average (MA), respectively.

- Autoregression (AR) (p): use of past observations in a time series to predict future values. The parameter 'p' indicates the number of lag observations included in the model.
- Integration (I) (d): Integration represents the differencing of observations to make the time series stationary. The parameter 'd' indicates the number of times differencing is performed.
- Moving Average (MA) (q): Moving Average involves modeling the error term as a linear combination of past error terms. The parameter 'q' indicates the size of the moving average window.

For this model, a monthly observation is taken, aggregating sales.

The chosen parameters (3, 0, 0) indicate an AR model with quarterly observations to capture trends and seasonality.

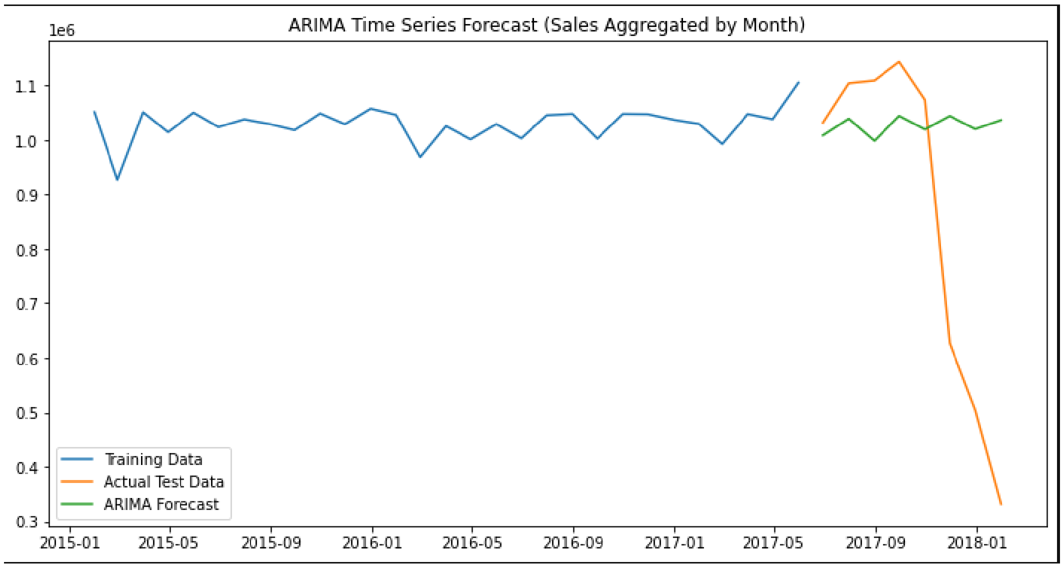


Figure 4.1 Forecasting Model

However, ensuring data stationarity is crucial for accurate predictions. The model highlights a significant drop in sales during the last couple of months, requiring further investigation of which is not covered in this project.

5 METRICS ANALYSIS

To measure accuracy metrics like MAE,MSE and RMSE can be used which show the relationship between the predicted and the observed values for the continuous models.

RMSE (Root Mean Squared Error):

RMSE measures the average magnitude of the errors between predicted and observed values.

The calculated RMSE for the ARIMA model is [347818.5624078372].

MSE (Mean Squared Error):

MSE provides a measure of the average squared differences between predicted and observed values.

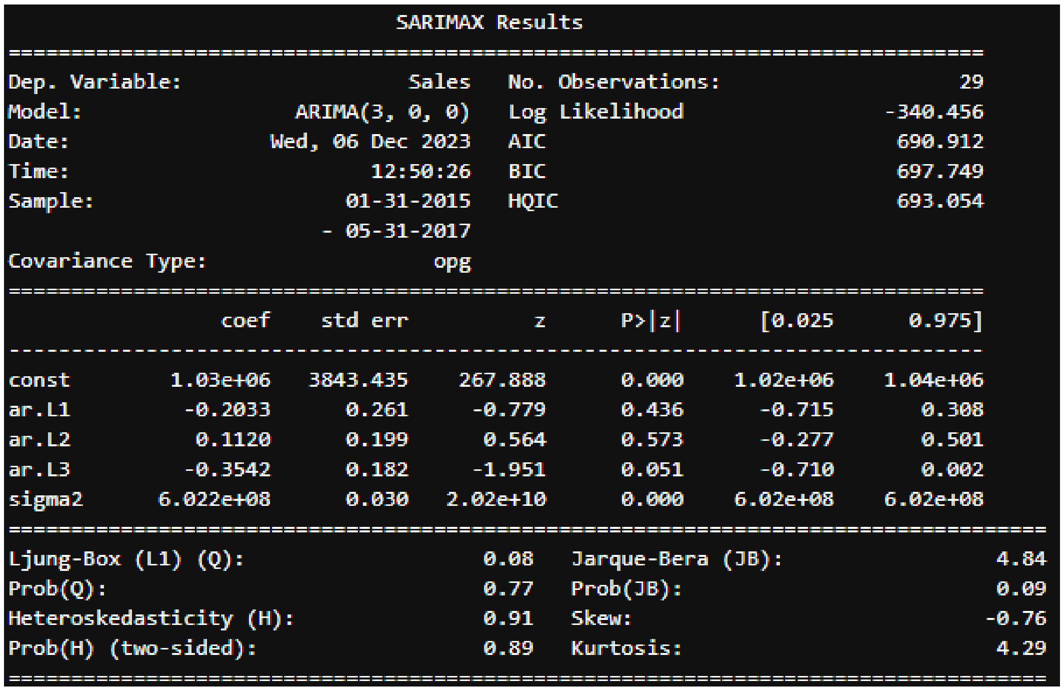
The calculated MSE for the ARIMA model is [120977752355.45453].

A way to distinguish between models is using the AIC (Akaike Information Criterion) and BIC (Bayesian Information Criterion) and log likelihood. AIC and BIC are model selection criteria that balance goodness of fit and model complexity. A lower AIC and BIC are indicators of a good model.

The calculated AIC and BIC for the ARIMA model are [690.912, 697.749] respectively.

Log Likelihood measures how well the model explains the observed data. A higher log likelihood is an indicator of a better model.

The Log Likelihood for the ARIMA model is [340.456]



```
                               SARIMAX Results
==============================================================================
Dep. Variable:                  Sales   No. Observations:                   29
Model:                 ARIMA(3, 0, 0)   Log Likelihood                -340.456
Date:                Wed, 06 Dec 2023   AIC                            690.912
Time:                        12:50:26   BIC                            697.749
Sample:                    01-31-2015   HQIC                           693.054
                         - 05-31-2017
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const        1.03e+06   3843.435    267.888      0.000    1.02e+06    1.04e+06
ar.L1         -0.2033      0.261     -0.779      0.436      -0.715       0.308
ar.L2          0.1120      0.199      0.564      0.573      -0.277       0.501
ar.L3         -0.3542      0.182     -1.951      0.051      -0.710       0.002
sigma2      6.022e+08      0.030   2.02e+10      0.000    6.02e+08    6.02e+08
===================================================================================
Ljung-Box (L1) (Q):                   0.08   Jarque-Bera (JB):                 4.84
Prob(Q):                              0.77   Prob(JB):                         0.09
Heteroskedasticity (H):               0.91   Skew:                            -0.76
Prob(H) (two-sided):                  0.89   Kurtosis:                         4.29
===================================================================================
```

Figure 5.1 Statistics Summary

The Ljung-Box test assesses the autocorrelation in the residuals of the time series model.

The Q statistic is [0.08] with a p-value of [0.77].

A high p-value suggests that there is insufficient evidence to reject the null hypothesis. Therefore, we accept the null hypothesis, indicating no autocorrelation in the residuals.

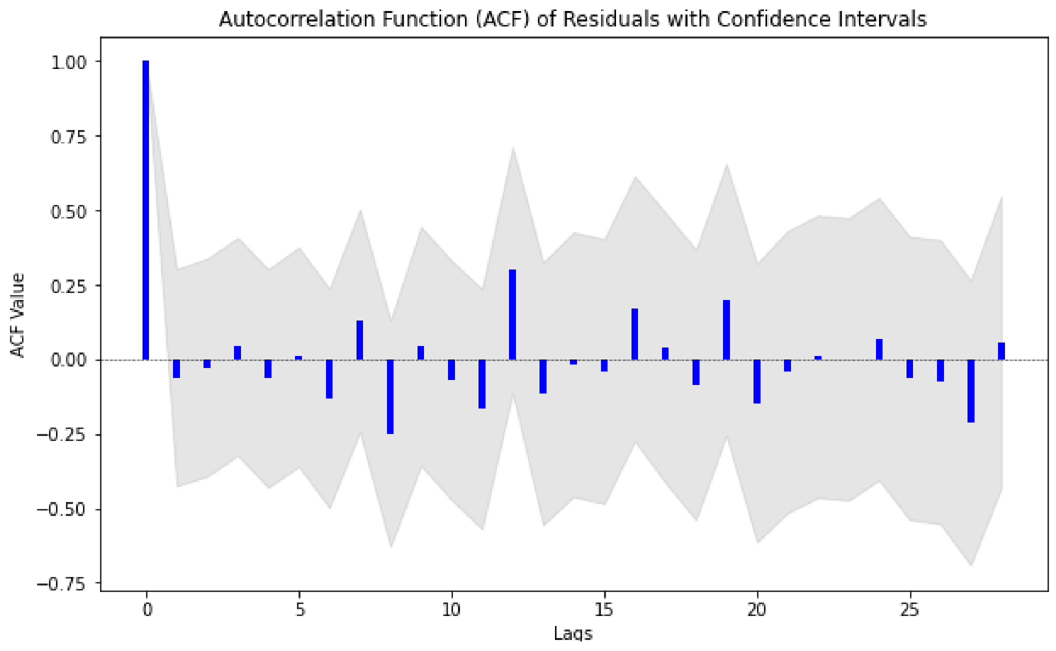


Figure 5.2 ACF Residuals

This outcome implies that the model has effectively captured the temporal trends and dependencies present in the data.

6 CONCLUSION

In conclusion, this project provides valuable insights and practice into the sales forecasting process within a supply chain domain. The ARIMA model, while indicating room for improvement, captures trends effectively. Recommendations include addressing delivery delays and further investigating the recent sales drop.

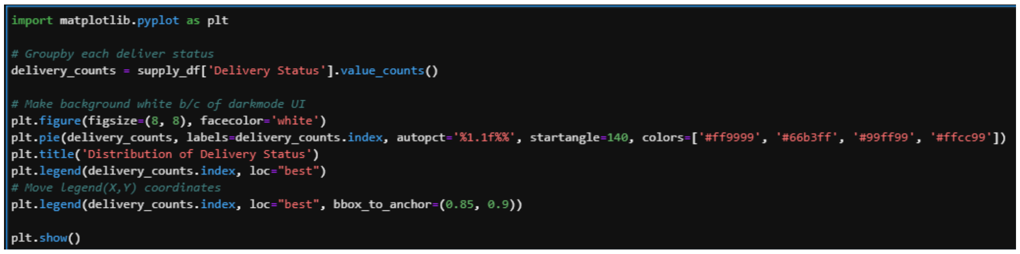
Areas for refinement encompass tackling the observed downturn in sales, exploring alternative models, and fine-tuning stationarity for more precise predictions. Overall, while the current model lays a solid foundation, ongoing adjustments and adaptability will be essential to align with the complexity of supply chain.

7 REFERENCES

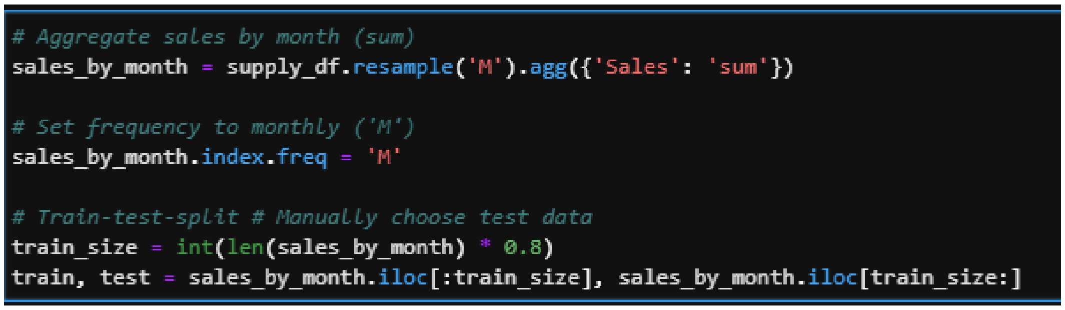
Link to data - https://www.kaggle.com/code/enjegodinez/supply-chain-demand-forecasting-prophet/input

8 APPENDICES

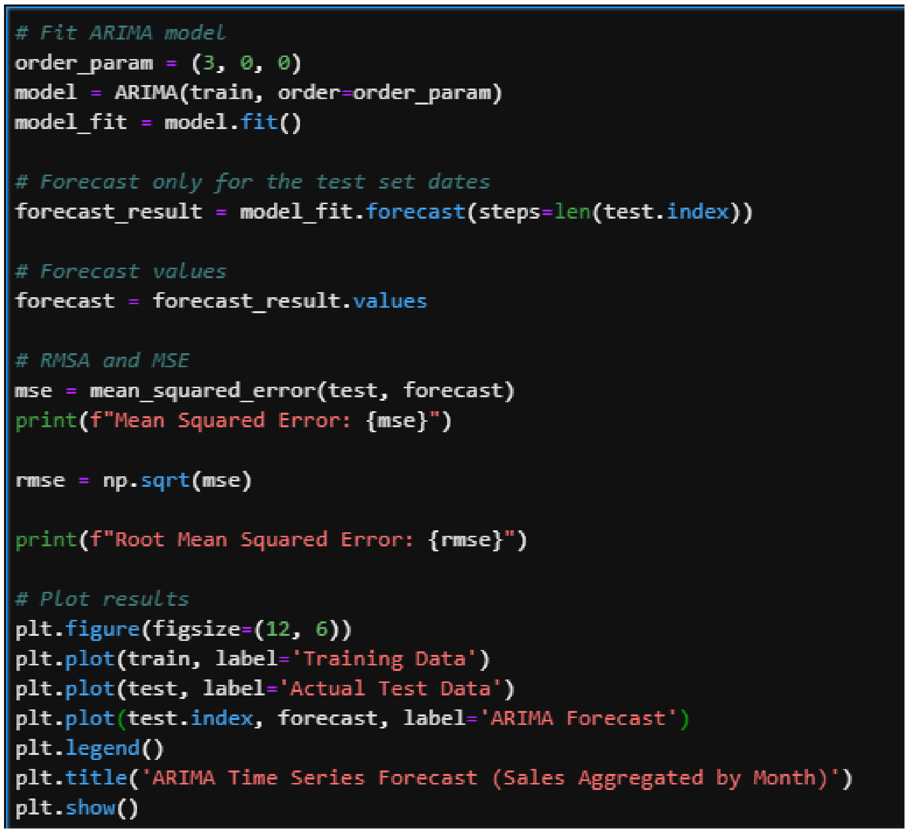
Screenshots of Parts of Python Script



```python
import matplotlib.pyplot as plt

# Groupby each deliver status
delivery_counts = supply_df['Delivery Status'].value_counts()

# Make background white b/c of darkmode UI
plt.figure(figsize=(8, 8), facecolor='white')
plt.pie(delivery_counts, labels=delivery_counts.index, autopct='%1.1f%%', startangle=140, colors=['#ff9999', '#66b3ff', '#99ff99', '#ffcc99'])
plt.title('Distribution of Delivery Status')
plt.legend(delivery_counts.index, loc="best")
# Move legend(X,Y) coordinates
plt.legend(delivery_counts.index, loc="best", bbox_to_anchor=(0.85, 0.9))

plt.show()
```



```python
# Aggregate sales by month (sum)
sales_by_month = supply_df.resample('M').agg({'Sales': 'sum'})

# Set frequency to monthly ('M')
sales_by_month.index.freq = 'M'

# Train-test-split # Manually choose test data
train_size = int(len(sales_by_month) * 0.8)
train, test = sales_by_month.iloc[:train_size], sales_by_month.iloc[train_size:]
```



```python
# Fit ARIMA model
order_param = (3, 0, 0)
model = ARIMA(train, order=order_param)
model_fit = model.fit()

# Forecast only for the test set dates
forecast_result = model_fit.forecast(steps=len(test.index))

# Forecast values
forecast = forecast_result.values

# RMSA and MSE
mse = mean_squared_error(test, forecast)
print(f"Mean Squared Error: {mse}")

rmse = np.sqrt(mse)

print(f"Root Mean Squared Error: {rmse}")

# Plot results
plt.figure(figsize=(12, 6))
plt.plot(train, label='Training Data')
plt.plot(test, label='Actual Test Data')
plt.plot(test.index, forecast, label='ARIMA Forecast')
plt.legend()
plt.title('ARIMA Time Series Forecast (Sales Aggregated by Month)')
plt.show()
```