D213: Advanced Data Analytics Task 1

Justin Huynh

Student ID: 012229514

M.S. Data Analytics

A1. Research Question

My research question for this project is "How can we predict future revenue trends and identify potential periods of declining revenue in the company's first two years of operation based on historical daily revenue data?"

A2. Objectives or Goals

For this analysis, we will have several objectives and goals that's defined below:

- Objective 1: Identify Long-Term Trends
    - Goal: Analyze the revenue data to identify and understand long-term trends over the two-year period. This includes determining whether revenue is generally increasing, decreasing, or stable over time.
- Objective 2: Detect Seasonal Patterns
    - Goal: Examine the data for any seasonal effects or recurring patterns (e.g. monthly or quarterly) that may influence revenue fluctuations. This can help us understand periods of high or low revenue that might correlate with specific customer behaviors or external factors.
- Objective 3: Forecast Future Revenue
    - Goal: Develop and apply time series forecasting models to predict future revenue based on historical data. Accurate forecasts can provide valuable insights for budgeting, resource allocation, and identifying potential periods of increased churn risk.
- Objective 4: Identify Anomalies or Unexpected Changes
    - Goal: Detect any unusual spikes or drops in revenue that may signal issues such as service outages, increased competition, or other

operational challenges. Understanding these anomalies can help in investigating potential causes and mitigating their impact.

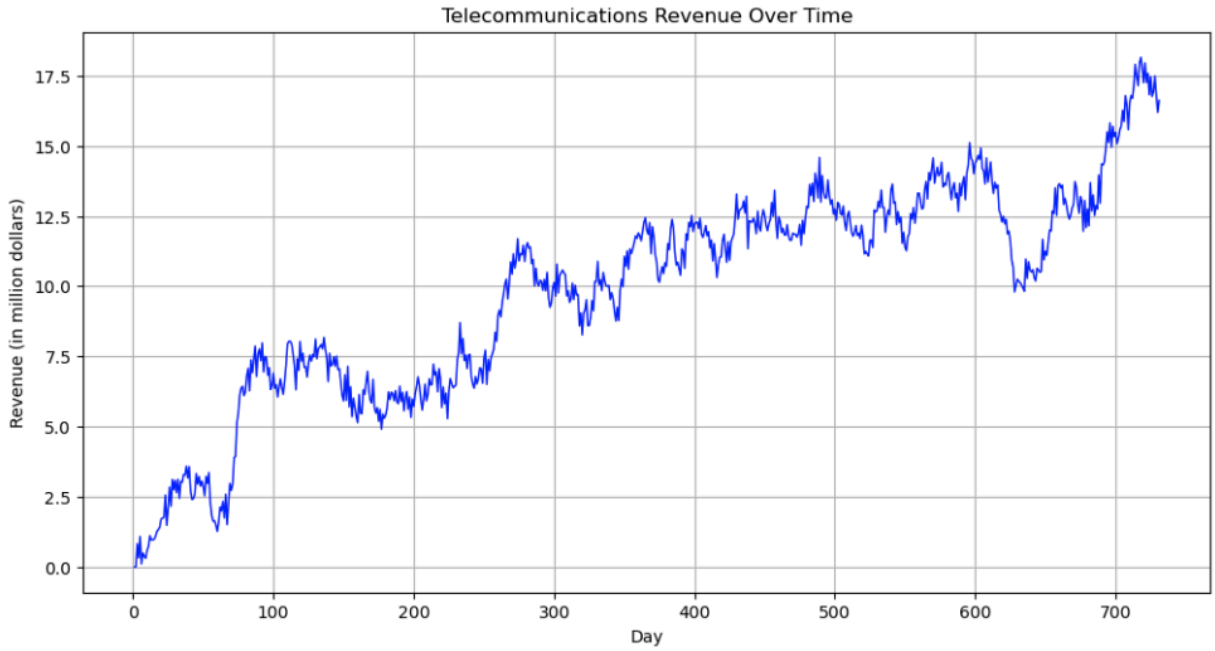
B. Summary of Assumptions

When working with time series data, a few assumptions should be considered to ensure the validity and effectiveness of the model. The two assumptions are stationarity and autocorrelation, which are both summarized below.

- Stationarity
    - A time series is considered stationary if its statistical properties, such as mean, variance, and autocovariance, do not change over time. In other words, the time series should have a consistent structure throughout its length, which is important for reliable forecasting. Stationarity is a crucial assumption for many time series models because these models rely on past data to predict future behavior. If the time series is not stationary, the model may produce inaccurate or biased forecasts.
- Autocorrelation
    - Autocorrelation refers to the correlation between a time series and a lagged version of itself. It measures the degree to which current values of the series are related to its past values. So autocorrelation allows time series models to capture relationships between current and past data points, which is crucial for making accurate predictions. It is fundamental in time series analysis because it indicates the presence of predictable patterns. Certain models like AutoRegressive and AutoRegressive

Integrated Moving Average use autocorrelation to predict future values based on past observations.

C1: Line Graph Visualization



See code attached in WGU_D213_Task_1.ipynb.

C2: Time Step Formatting

The time series is structured with each row representing a single day over a two-year period. The data is sequentially ordered by day, with no explicit gaps in the measurement. There are no missing days or gaps in the data. Each day from the first to the last is accounted for, providing a continuous record of revenue. The sequence spans 731 days, representing exactly two years of daily revenue data.

C3: Stationarity

To evaluate the stationarity of the time series, we'll use the Augmented Dickey-Fuller (ADF) test. This test helps determine whether a time series is stationary

by checking if there is a unit root present in the data.

```
ADF Statistic: -1.9246121573101873
p-value: 0.32057281507939484
Critical Values:
   1%: -3.4393520240470554
   5%: -2.8655128165959236
   10%: -2.5688855736949163
```

The ADF statistic is greater than the critical values at all significance levels (1%, 5%, and 10%), and the p-value is greater than 0.05. This means the time series is not stationary. This suggests that the mean and variance of the revenue data may be changing over time, making it necessary to apply transformations (differencing) to achieve stationarity before applying certain time series models.

See code attached in WGU_D213_Task_1.ipynb.

C4: Steps to Prepare the Data

To prepare the data for time series analysis, we'll follow these steps:

- Since the time series is not stationary, we'll apply differencing to the revenue data to stabilize the mean. This is a common approach to make a series stationary.
- Training Set: We'll use the first part of the data (e.g., the first 80% of days) to train the time series models.
- Test Set: The remaining data (e.g., the last 20%) will be used to evaluate the model's performance and validate its forecasting accuracy.

```
# apply first-order differencing
teleco_time_series['Revenue_diff'] = teleco_time_series['Revenue'].diff().dropna()

# drop NaN values created by differencing
teleco_time_series_diff = teleco_time_series.dropna()

# split the data (80% training, 20% testing)
split_point = int(len(teleco_time_series_diff) * 0.8)
train_set = teleco_time_series_diff[:split_point]
test_set = teleco_time_series_diff[split_point:]

# display the first few rows of the training set
train_set.head()
```

	Day	Revenue	Revenue_diff
1	2	0.000793	0.000793
2	3	0.825542	0.824749
3	4	0.320332	-0.505210
4	5	1.082554	0.762222
5	6	0.107654	-0.974900

See code attached in WGU_D213_Task_1.ipynb.

C5: Prepared Data Set

Copies of the fully prepared data sets will be submitted as

- 'cleaned_teleco_time_series_train_d213_task1.csv'
- 'cleaned_teleco_time_series_test_d213_task1.csv'

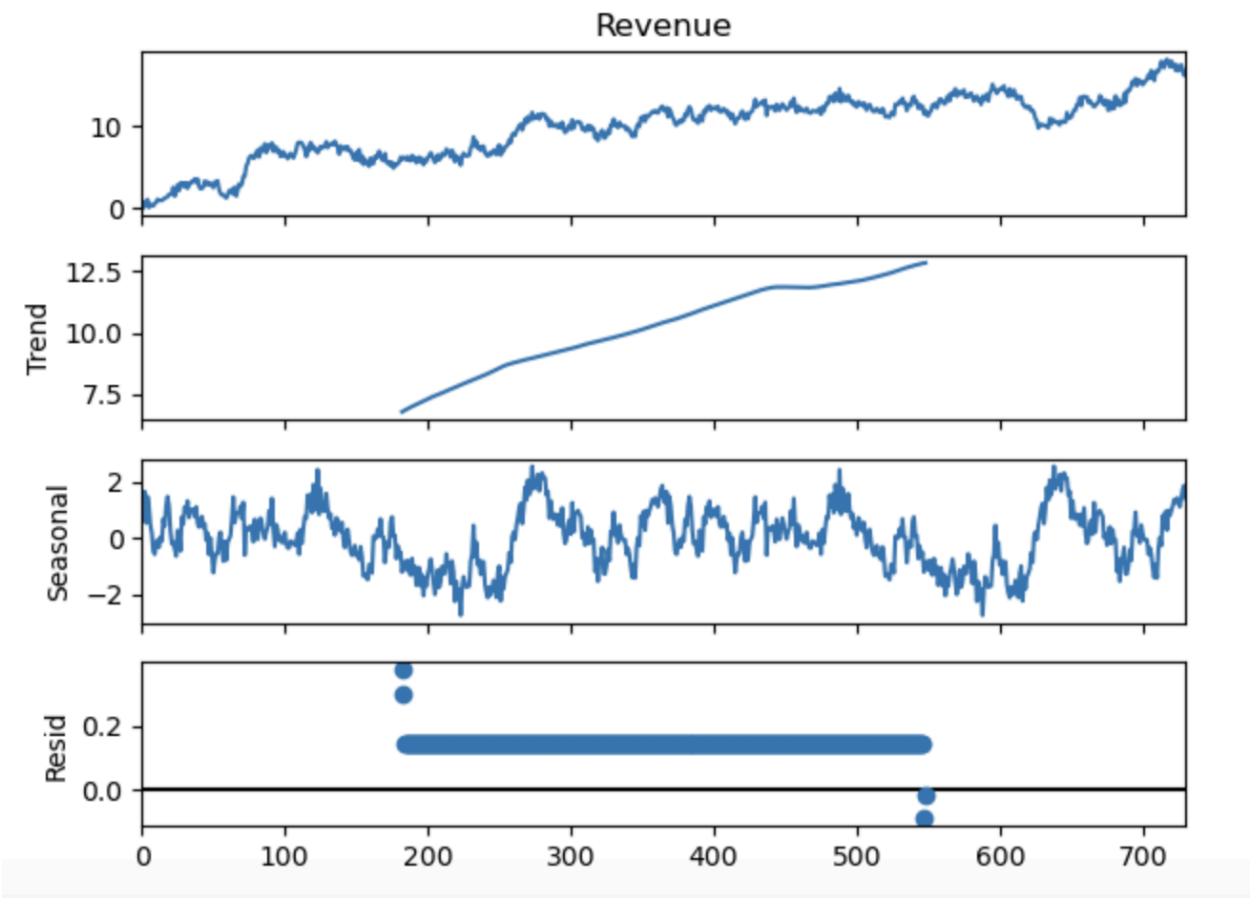
D1: Report Findings and Visualizations

Annotated findings with visualizations for each of the 6 elements:

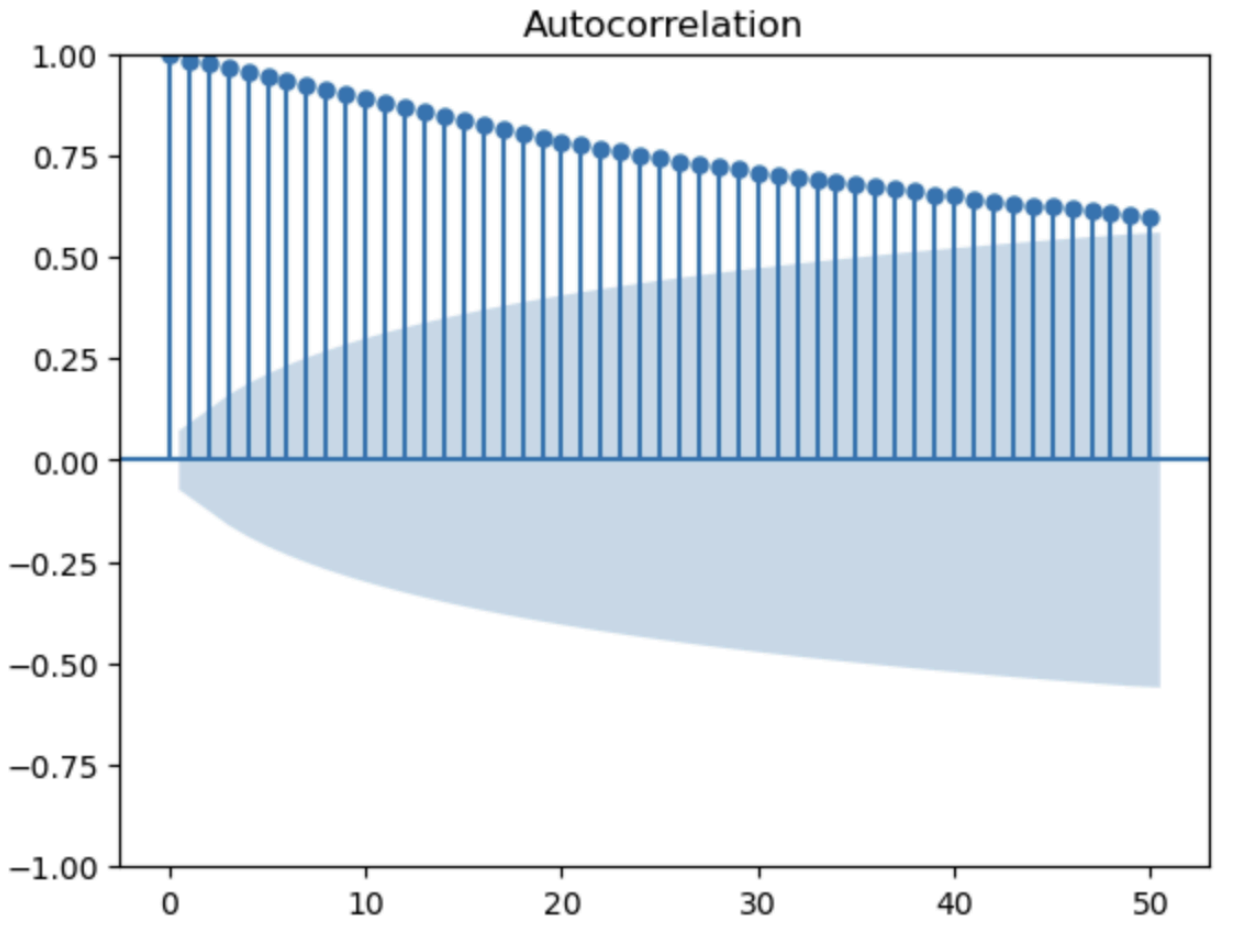
```
# decompose the full dataset
decomposition = seasonal_decompose(teleco_time_series['Revenue'], model='additive', period=365)
decomposition.plot()
plt.show()

# check the seasonal component
seasonal_component = decomposition.seasonal

# trends
trend_component = decomposition.trend
```

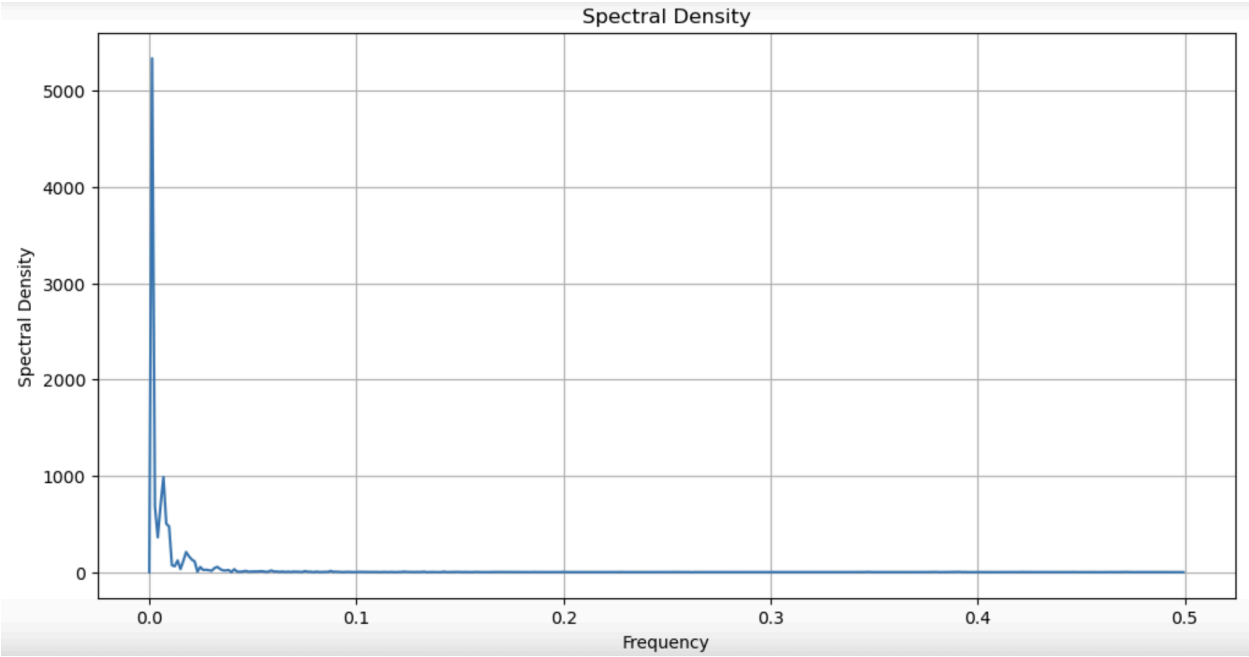



```
# plot the ACF for the original revenue data
plot_acf(teleco_time_series['Revenue'].dropna(), lags=50)
plt.show()
```



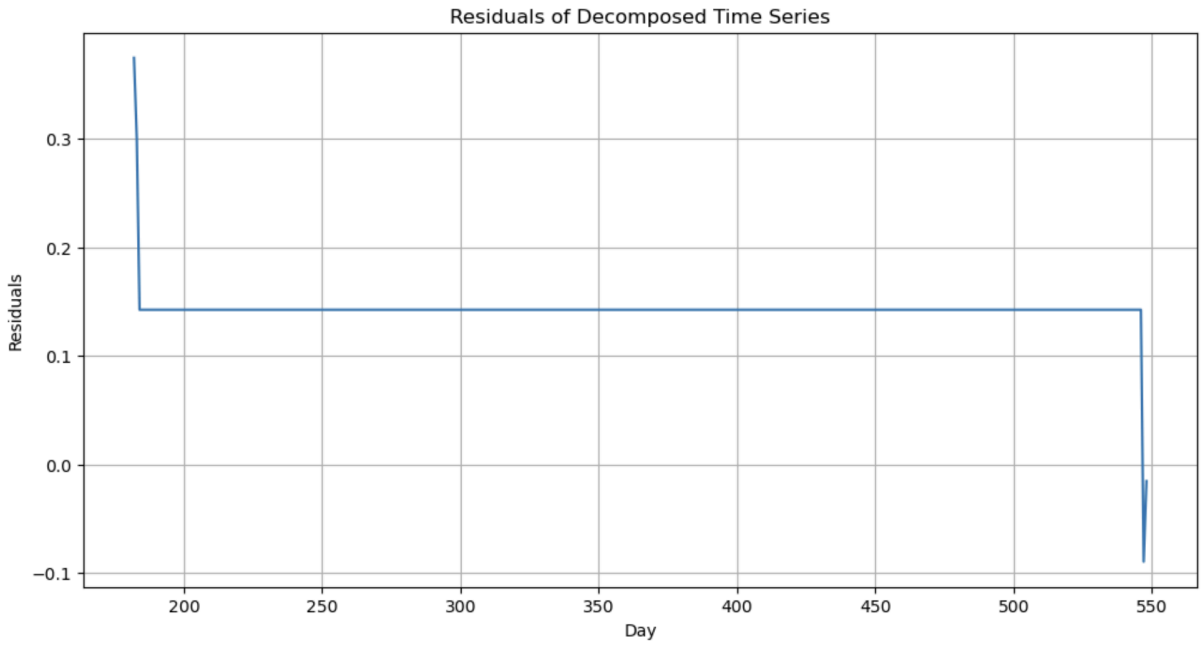

```
# compute the periodogram
frequencies, spectrum = periodogram(teleco_time_series['Revenue'].dropna())

# plot the spectral density
plt.figure(figsize=(12, 6))
plt.plot(frequencies, spectrum)
plt.title('Spectral Density')
plt.xlabel('Frequency')
plt.ylabel('Spectral Density')
plt.grid(True)
plt.show()
```




```
# confirming lack of trends in the residuals
# plot residuals from decomposition
plt.figure(figsize=(12, 6))
plt.plot(decomposition.resid)
plt.title('Residuals of Decomposed Time Series')
plt.xlabel('Day')
plt.ylabel('Residuals')
plt.grid(True)
plt.show()

# perform an ADF test on residuals to confirm stationarity
from statsmodels.tsa.stattools import adfuller
adf_result_resid = adfuller(decomposition.resid.dropna())
print(f'Residuals ADF Statistic: {adf_result_resid[0]}')
print(f'Residuals p-value: {adf_result_resid[1]}')
```



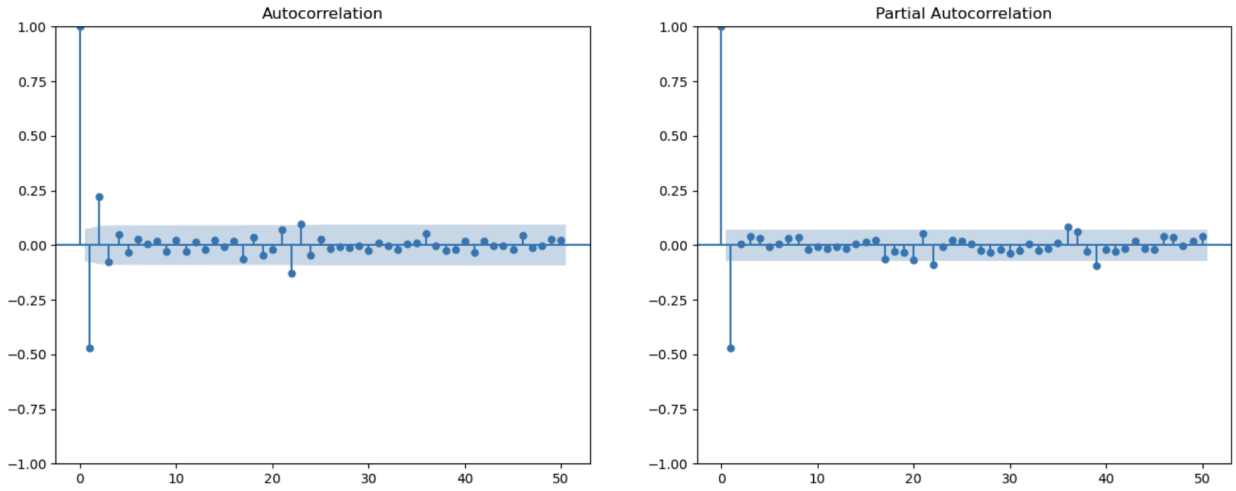

```
Residuals ADF Statistic: -12.362021927319837
Residuals p-value: 5.519275932513205e-23
```

See code attached in WGU_D213_Task_1.ipynb.

D2: Arima Model

```
# differencing to make the series stationary
teleco_time_series['Revenue_diff'] = teleco_time_series['Revenue'].diff().dropna()

# plot the ACF and PACF
fig, ax = plt.subplots(1, 2, figsize=(16, 6))
plot_acf(teleco_time_series['Revenue_diff'].dropna(), lags=50, ax=ax[0])
plot_pacf(teleco_time_series['Revenue_diff'].dropna(), lags=50, ax=ax[1])
plt.show()
```




```python
# fit the ARIMA(1,1,1) model
model = ARIMA(teleco_time_series['Revenue'], order=(1, 1, 1))
model_fit = model.fit()

# summary of the model
print(model_fit.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                Revenue   No. Observations:                  731
Model:                 ARIMA(1, 1, 1)   Log Likelihood                -490.326
Date:                Tue, 03 Sep 2024   AIC                            986.652
Time:                        01:21:21   BIC                           1000.431
Sample:                             0   HQIC                           991.968
                                  - 731
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.4804      0.068     -7.063      0.000      -0.614      -0.347
ma.L1          0.0176      0.077      0.227      0.821      -0.134       0.169
sigma2         0.2243      0.013     17.764      0.000       0.200       0.249
===================================================================================
Ljung-Box (L1) (Q):                   0.02   Jarque-Bera (JB):                 2.09
Prob(Q):                              0.89   Prob(JB):                         0.35
Heteroskedasticity (H):               1.02   Skew:                            -0.02
Prob(H) (two-sided):                  0.88   Kurtosis:                         2.74
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

The full dataset was used for model training because the ARIMA model is inherently designed to model and forecast based on the entire available historical data. In cases where the dataset is relatively small (in this case, 731 observations), using the full dataset allows the model to capture all available patterns and dynamics. Since ARIMA models focus on leveraging past patterns to predict future values, using the entire time dataset ensures that the model has access to the most complete historical context. By using the full dataset, The ARIMA model can better capture the autocorrelations and moving averages present across the entire time series. This improves the model's ability to accurately represent long-term patterns. Also, since the dataset only spans 731 days, splitting the data even further could reduce the model's accuracy due to less data being available to learn from, which is important for time series forecasting.

The ARIMA(1, 1, 1) order was selected based on the analysis of the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots after differencing the series to make it stationary.

ARIMA(1, 1, 1) Explanation:

- p = 1 (AR Term)
    - The PACF plot shows a significant spike at lag 1, after which the correlation quickly drops off. This pattern suggests that an AR(1) process is sufficient. The AR(1) term captures the dependency of the current value on the immediately preceding value, making this an optimal choice for the model.
- d = 1 (Differencing)

    - The differencing term (d=1) was chosen because the original revenue data showed trends and non-stationarity. After applying one difference (diff()), the time series became stationary as confirmed by stationarity tests like the Augmented Dickey-Fuller (ADF) test. So d=1 is the appropriate choice for differencing to remove trends and achieve stationarity.

- q = 1 (MA Term)
    - The ACF plot shows a significant spike at lag 1, after which the autocorrelations quickly diminish. This pattern indicates that an MA(1) process is optimal. The MA(1) term captures the dependency of the current value on the immediate past forecast errors (residuals), making this order suitable for the model.

The reason ARIMA(1, 1, 1) is optimal is because of model simplicity. It creates a balance between capturing enough complexity in the data without overfitting or adding unnecessary parameters. Adding more AR or MA terms would most likely result in a more complex model without significantly improving accuracy. Another reason this model is optimal is because of AIC and BIC Scores. The Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) values of the model were relatively low, indicating that the model provides a good fit to the data without being overly complex. Lower AIC and BIC values suggest the model is statistically efficient and avoids overfitting.

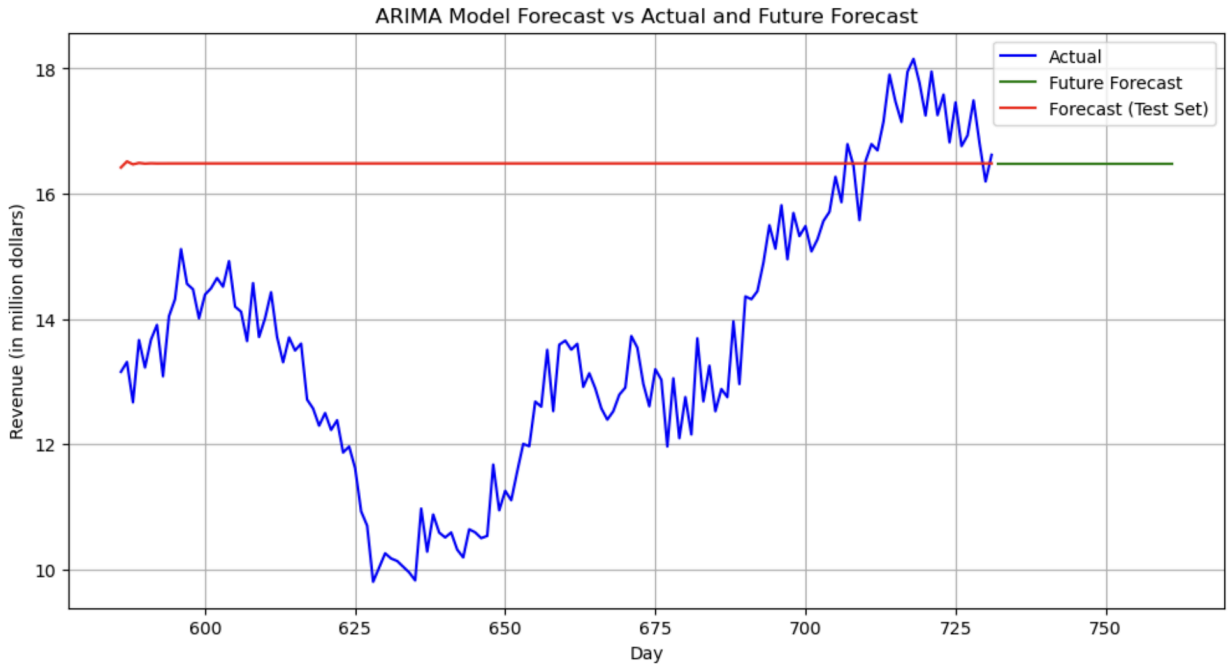
See code attached in WGU_D213_Task_1.ipynb.

D3: Forecasting Using Arima Model

```
# define how many periods into the future to forecast (30 days into the future)
future_steps = 30
total_forecast_steps = len(test_set) + future_steps

# forecast for both the test set and into the future
forecast = model_fit.forecast(steps=total_forecast_steps)

# plot the forecast against the actual test set values and future forecast
plt.figure(figsize=(12, 6))
plt.plot(test_set['Day'], test_set['Revenue'], label='Actual', color='blue')
plt.plot(range(test_set['Day'].max() + 1, test_set['Day'].max() + future_steps + 1), forecast[-future_steps:], label
plt.plot(test_set['Day'], forecast[:len(test_set)], label='Forecast (Test Set)', color='red')
plt.title('ARIMA Model Forecast vs Actual and Future Forecast')
plt.xlabel('Day')
plt.ylabel('Revenue (in million dollars)')
plt.legend()
plt.grid(True)
plt.show()
```



See code attached in WGU_D213_Task_1.ipynb.

D4: Output and Calculations

All output and calculations of the analysis are provided above in D1-D3.

See code attached in WGU_D213_Task_1.ipynb.

D5: Code

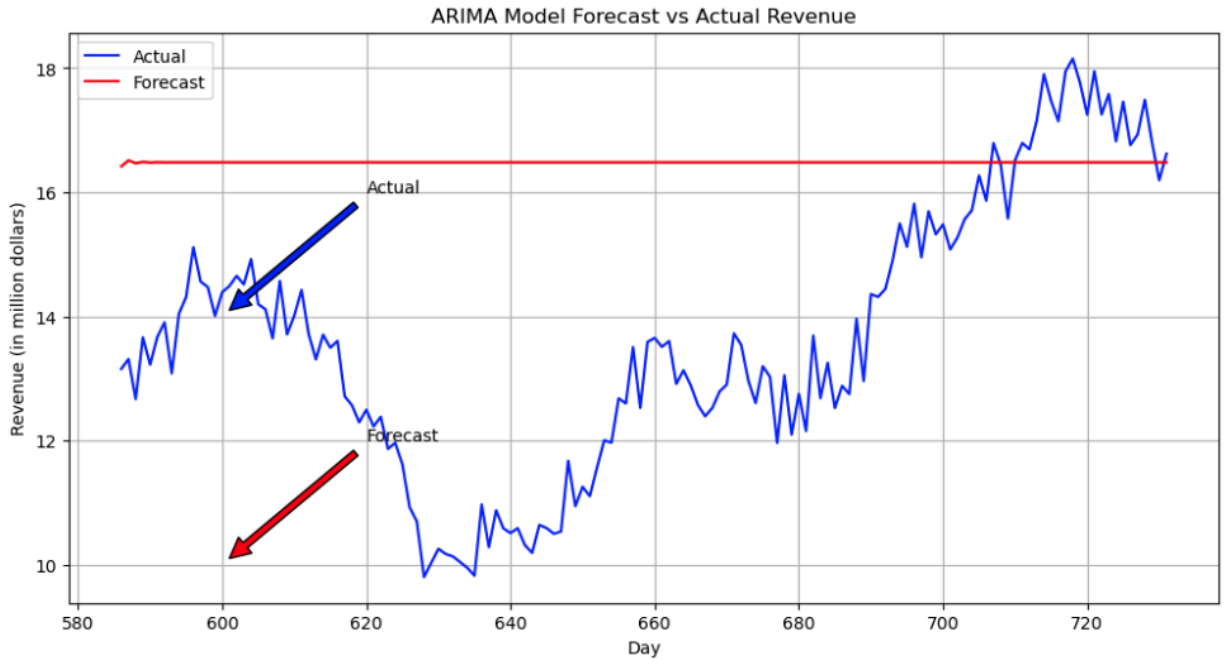
See code attached in WGU_D213_Task_1.ipynb.

E1: Results

The selected ARIMA(1, 1, 1) model was based on an analysis of the ACF and PACF plots that indicates an autoregressive process with one lag and a moving average process with one lag. The model's differencing order (d=1) was applied to ensure stationarity in the data. The AR term (AR(1)) was found to be significant while the MA term (MA(1)) was not, indicating that the autoregressive component had a more substantial effect on forecasting than the moving average term. The prediction interval provided a 95% confidence range around the forecasted values but the model's forecast was relatively flat. The length of the forecast was justified by using the 146-day test set period to give a practical window for comparing predictions with actual outcome. The model evaluation relied on both quantitative (Mean Absolute Error) and visual assessments. This indicates that while the model captured general trends, it struggled with periods of high variability in the data.

E2: Annotated Visualization

The visualization shows a comparison between the actual revenue (blue line) and the forecasted revenue (red line) over the 146-day test period. The ARIMA model predicted a relatively flat revenue trajectory, with the forecast failing to capture significant fluctuations in the actual data, like the sharp decline around day 600. The forecast line remained steady, which indicates the model to expect stable revenue even as the actual values fluctuate. The graph highlights key moments where the forecast deviated from the actual performance. This gap suggests that the model may not be adequately capturing the volatility in the time series, and that future models might need to address these fluctuations more effectively.



See code attached in WGU_D213_Task_1.ipynb.

E3: Recommendation

Based on the results, the ARIMA(1, 1, 1) model is useful for predicting overall revenue trends but doesn't quite capture the periods of volatility, such as sudden drops and increases in the trend. So it is recommended that a company should adopt a cautious approach to decision-making, especially in periods where revenue is expected to decline. To improve forecast accuracy, it would be a good idea to consider a SARIMAX model that accounts for seasonality or incorporates external factors like marketing campaigns or customer churn rates that may explain these sharp revenue changes. Continuous monitoring is also recommended for model refinement as more data becomes available to ensure that the forecasts align with the company's needs.

F: Reporting

The files submitted with this task will be labeled as

- 'WGU_D213_Task_1.html'

- ‘WGU_D213_Task_1.ipynb’

G: Sources for Third-Party Code

1. “Decomposing Time Series Data” Retrieved from https://www.kaggle.com/code/chanakyavivekkapoor/decomposing-time-series-data
2. “Time Series: Interpreting ACF and PACF” Retrieved from https://www.kaggle.com/code/iamleonie/time-series-interpreting-acf-and-pacf
3. “Scipy.signal. Periodogram” Retrieved from https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.periodogram.html

H: Sources

1. “Guide to Time-Series Analysis with Python” Retrieved from https://www.timescale.com/blog/how-to-work-with-time-series-in-python/
2. “ARIMA Model Explained | Time Series Forecasting” Retrieved from https://www.youtube.com/watch?v=-_2wOrEuFaM
3. “How to Build ARIMA Model in Python for time series forecasting?” Retrieved from https://www.projectpro.io/article/how-to-build-arima-model-in-python/544