12/13/2023

**Indrani Eagapati**

**STAT-773**

**AMZON STOCK PRICES ANALYSIS**

Project report

**Table of Contents:**

* Abstract
* Introduction
* Analysis
* Conclusion
* Future Work

**Abstract:**

This project conducts an important exploration into the historical stock price data of Amazon, a leading global technology and e-commerce giant. As a significant player in the stock market, Amazon's stock prices are of keen interest to investors and analysts seeking to make informed decisions and manage risks effectively. Through a focused time, series analysis, this research aims to uncover underlying patterns and trends within Amazon's stock prices, offering valuable insights for strategic decision-making.

**Introduction:**

The analysis focuses on understanding and predicting trends in financial markets using historical stock price data. The dataset is sourced from Yahoo Finance[1].This study is performed on the closing values of the monthly stock price of Amazon from 2007 to 2021. The monthly stock price from 2022 is used as a validation set. The statistics of the stock price is as follows.

**Statistics**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Variable** | **N** | **N\*** | **Mean** | **SE Mean** | **StDev** | **Minimum** | **Q1** | **Median** | **Q3** | **Maximum** |
| Adj Close | 180 | 0 | 43.69 | 3.79 | 50.79 | 1.88 | 7.96 | 16.94 | 75.49 | 175.35 |

**Analysis:**

A graph showing a line graph

Description automatically generated

The time series plot of the closing values of the monthly stock prices of amazon stock seems to have an increasing trend which seems to be quadratic or exponential in the nature. There is a notable spike in the ad close of the time series (around index 163). This spike is likely due to some external event. It is worth noting that there are some spikes in the middle of the series as well.

A graph with a red line and blue line

Description automatically generated

The Auto-Correlation Function (ACF) plot is used to visualize the correlation between a time series and its lagged values. As we can see above, the ACF values are high and slowly decaying to zero, which shows that the time series at hand has some kind of trend. Both the time series and ACF plot shows that there is no seasonality yearly but there is a significant cyclic behavior.

A graph with a red line

Description automatically generatedA graph with a red line and blue line

Description automatically generated

The trend analysis plot using quadratic model and exponential shows that exponential model is performing better on this data with MAPE of 12.33 when compared to the quadratic model with MAPE of 65.381. A higher MAPE suggests that the quadratic model has a larger percentage difference between its predicted values and the actual values when compared to the exponential model. This confirms that the trend is exponential.

**Regression Model:**

Given the established exponential nature of the trend, the recommended approach is to employ a log transformation followed by the application of linear regression.

A graph showing a line

Description automatically generated

We can see the time series plot after applying log and the trend looks fairly linear.

* The regression with the index as continuous predictor resulted in the Durbin-Watson statistic value of 0.26 which indicates strong positive autocorrelation between the errors(1.1.1).
* Including a lag term in the model, as it can help capture and account for any temporal patterns or dependencies in the could improve the Durbin Watson value.
* The regression model with the lag term has improved the Durbin Watson statistic but the residuals from the model do not follow normal distribution, which is an assumption for the regression model(1.1.2).
* Introducing another lag term could improve the model further. Let’s look at this model:

**Best regression model:**

**Regression Equation**

|  |  |  |
| --- | --- | --- |
| log(Adj Close) | = | 0.1497 + 0.9949 log(Adj Close)\_Lag1 + 0.003687 index - 0.1481 log(Adj Close)\_Lag1\_Lag1 |

**Model Summary**

|  |  |  |  |
| --- | --- | --- | --- |
| **S** | **R-sq** | **R-sq(adj)** | **R-sq(pred)** |
| 0.0900129 | 99.50% | 99.50% | 99.47% |

**Analysis of Variance**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Source** | **DF** | **Adj SS** | **Adj MS** | **F-Value** | **P-Value** |
| Regression | 3 | 283.006 | 94.3353 | 11643.01 | 0.000 |
| log(Adj Close)\_Lag1 | 1 | 1.422 | 1.4216 | 175.45 | 0.000 |
| index | 1 | 0.123 | 0.1233 | 15.22 | 0.000 |
| log(Adj Close)\_Lag1\_Lag1 | 1 | 0.032 | 0.0318 | 3.92 | 0.049 |
| Error | 174 | 1.410 | 0.0081 |  |  |
| Total | 177 | 284.416 |  |  |  |

**Durbin-Watson Statistic**

|  |  |
| --- | --- |
| Durbin-Watson Statistic = | 1.99761 |

The p-values of log(Adj Close)\_Lag1, index and log(Adj Close)\_Lag1\_Lag1 are all less than 0.05 which shows that all the terms in the model are significant in predicting the values of log(Adj Close)The critical values of Durbin-Watson statistic are [1.728,1.820] for n=180 and 3 predictors. The value of our regression model is 1.99 which is slightly higher than the critical value but not by a lot. This could mean that the residuals are negatively auto correlated.

A graph of a normal plot

Description automatically generated

We can see that the residuals are normally distributed according to Anderson test with p-value of 0.109 which is greater than 0.05.

A graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generated

Residual vs fitted values shows random scatter of points which meets the assumption of regression. The spread of residuals should be relatively constant across all levels of the fitted values. On average, the residuals are centered around zero.

**Forecasts:**

A screenshot of a table

Description automatically generated

A graph of different colored lines

Description automatically generated

The observed values generally fall within the lower and upper bounds of the forecasts, indicating that the forecasting model successfully captures the overall data trend. Nonetheless, there are 9 out of 12 points where observed values deviate significantly from the forecasts, likely attributed to unaccounted factors in the model. Accuracy in forecasting is more pronounced for recent time periods, as the model is trained on historical data and the most recent data tends to be a more accurate representation of the current situation. Applying an exponential model appears to result in predicted values that exhibit substantial growth compared to the observed values. This suggests the potential need for employing two distinct models to achieve more accurate predictions.

Top of Form

Bottom of Form

**Exponential smoothing:**

Double Exponential Smoothing, also known as Holt's method, is a time series forecasting technique that extends the simple exponential smoothing method to handle data with trends. Double exponential smoothing is designed to capture and forecast data with a linear trend. As we have already established that the trend in this case is exponential, we are applying double exponential smoothing on the log transformed values.

Using python script which reads time series data from an Excel file, assumes that the data has already been log-transformed, and initializes a set of candidate values for the smoothing parameters alpha and beta. It then iterates through all combinations of alpha and beta values ranging from 0.1 to 0.9, fits a double exponential smoothing model to the data using the specified parameters, calculates the Mean Absolute Percentage Error (MAPE) on the fitted values, and keeps track of the best alpha, beta, and corresponding MAPE. The results are stored in lists for later plotting, and a 3D surface plot is generated to visualize the MAPE values for different alpha and beta combinations. Finally, the code prints the best alpha, beta i.e. which has the lowest MAPE. The purpose of this code is to identify the optimal alpha and beta values that minimize forecasting error for the given time series data.(1.1.8).

The results are as follows:

A diagram of a graph

Description automatically generated

After verifying that these are indeed the best level and trend values by experimenting with different values in Minitab(1.1.3,1.1.4). The best double exponential smoothing method is:

**Smoothing Constants**

|  |  |
| --- | --- |
| α (level) | 0.9 |
| γ (trend) | 0.1 |

**Accuracy Measures**

|  |  |
| --- | --- |
| MAPE | 3.69171 |
| MAD | 0.07445 |
| MSD | 0.00977 |

A graph with red and blue lines

Description automatically generated

**Forecasts**:

: A screenshot of a data

Description automatically generated

A graph of a number of graphs

Description automatically generated

The double exponential smoothing model yields forecasts that closely align with the observed values, with all data points falling within the prediction bounds. This indicates the model's aptness for the provided data. It is essential to note that double exponential smoothing is most effective for predicting immediate time indices, and its suitability diminishes when forecasting for indices further into the future.

**ARIMA:**

A graph with a number of lines

Description automatically generatedA graph with a line graph

Description automatically generated

Looking at the ACF or PACF plots of first order difference, it is not possible to find the definitive order of moving averages or auto regressive as the order can be determined by the number of number of after which the ACF and PACF values becomes zero. We can observe from the above that the ACF and PACF values are high in between, this may be due to the usual highs and dips of the series in the middle.

A graph with a line graph

Description automatically generated

A graph with a line graph

Description automatically generated with medium confidence

Looking at the ACF and PACF plot of 2nd order difference, moving average of order 1 as ACF values drop to zero after lag 1 and auto regressive of order 3, As PACF values drop to zero after 3 lags. Box-pierce p-values are less than 0.05 which indicates that the residuals are not stationary(1.1.5). After looking at some more ARIMA models, still the box-pierce p-values are not improving, this may be due to the variability in the series that we established in the start(1.1.6)(1.1.7). Now let’s look at the log transformation and then try to fit the ARIMA model.

A graph with a line graph

Description automatically generatedA graph with a line graph

Description automatically generated

A graph with a line graph

Description automatically generatedA graph with a line graph

Description automatically generated

Looking at the ACF and PACF plots, differencing order of 2, moving average order of 1 and auto-regressive order of 0 seems to fit the data well based on above plots. We can see that the first order difference plot is not conclusive in determining the moving average and auto-regressive order. In the second order difference of the log values of our series, we can see that the PACF values are decreasing slowly, whereas as ACF drop to zero after 1st lag which indicates MA of order 1.

**Final Estimates of Parameters**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type** | **Coef** | **SE Coef** | **T-Value** | **P-Value** |
| MA   1 | 0.994154 | 0.000359 | 2768.58 | 0.000 |
| Constant | -0.000005 | 0.000137 | -0.04 | 0.968 |

**Residual Sums of Squares**

|  |  |  |
| --- | --- | --- |
| **DF** | **SS** | **MS** |
| 176 | 1.56035 | 0.0088656 |

Back forecasts excluded

**Model Summary**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **DF** | **SS** | **MS** | **MSD** | **AICc** | **AIC** | **BIC** |
| 177 | 1.56380 | 0.0088350 | 0.0087854 | -329.649 | -329.718 | -323.354 |

MS = variance of the white noise series

**Modified Box-Pierce (Ljung-Box) Chi-Square Statistic**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lag** | **12** | **24** | **36** | **48** |
| Chi-Square | 9.37 | 19.19 | 26.83 | 34.58 |
| DF | 10 | 22 | 34 | 46 |
| P-Value | 0.497 | 0.634 | 0.804 | 0.892 |

Forecasts:

A screenshot of a table

Description automatically generated

A graph of a number of points

Description automatically generated with medium confidence

The scatterplot above displays the forecasts with prediction limits alongside the observed values. Notably, the Arima model's bounds fail to encompass 5 out of the 12 observed values. This suggests that our ARIMA model tends to overestimate the data, possibly attributed to the underlying exponential assumption.

**Bootstrap:**

Bootstrapping is employed in situations where no assumptions can be made about the available data. In this approach, 1000 resamples are generated by resampling using residuals. As we have seen that the using log values is leading to overestimating the values, we are going to apply this technique on the stock values instead of log of stock values as above. The model incorporates residual errors derived from the model, which are then added to the fitted values obtained from the model. Additionally, double exponential smoothing is integrated into this model.

**Forecast bounds**:

A screenshot of a table

Description automatically generated

A graph of a number of values

Description automatically generated

Observing the results of bootstrapping with 1000 samples, it becomes evident that 5 out of the 12 data points are not captured. Despite this, the overall trend appears to align with the dataset pattern.

**Comparing above models:**

**Model Summary Regression**

|  |  |  |  |
| --- | --- | --- | --- |
| **S** | **R-sq** | **R-sq(adj)** | **R-sq(pred)** |
| 0.0900129 | 99.50% | 99.50% | 99.47% |

Bottom of Form

**A close up of a number

Description automatically generated**

**Model Summary ARIMA:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **DF** | **SS** | **MS** | **MSD** | **AICc** | **AIC** | **BIC** |
| 177 | 1.56380 | 0.0088350 | 0.0087854 | -329.649 | -329.718 | -323.354 |

MS = variance of the white noise series

**Accuracy Measures Exponential smoothing**

|  |  |
| --- | --- |
| MAPE | 3.69171 |
| MAD | 0.07445 |
| MSD | 0.00977 |

The sum of squares for the regression model is 0.05, while for the ARIMA model, it is 1.5, indicating a slightly better performance of the regression model. When comparing the Mean Squared Deviation (MSD) between the ARIMA and Double Exponential Smoothing models, the MSD for ARIMA (0.00878) is marginally lower than that of Double Exponential Smoothing (0.00977), suggesting superior performance for the ARIMA model. In an overall comparison of MSD values, Regression exhibits the lowest MSD at 0.079, followed by ARIMA and Double Exponential Smoothing in ascending order of performance.

**Conclusion**:

In the culmination of our time series analysis on Amazon stock prices, various models such as regression on log transformation, exponential smoothing, ARIMA, and bootstrap with 1000 samples were applied. When assessing the forecast for the next 12 periods and comparing it with the observed values, a notable finding emerged: all observed values fell within the prediction bounds of the exponential model. Conversely, discrepancies were noted in other models, where some observed values deviated from the predictions. This compelling evidence strongly indicates that double exponential smoothing stands out as the most suitable model for accurately capturing and predicting the behavior of Amazon stock prices in our dataset. It is important to highlight, however, that exponential smoothing methods demonstrate optimal performance for immediate next intervals rather than over extended time intervals. In such cases, regression or ARIMA models may offer better predictive capabilities.

**Future Work:**

Future work in the field of time series analysis for Amazon stock prices should consider incorporating external factors that influence stock prices. It is well-established that various external variables play a crucial role in shaping stock trends. The presence of outliers in our models may be attributed to the omission of these influential factors. To enhance the accuracy and robustness of our models, future research could explore integrating external variables such as economic indicators, market trends, and company-specific events into the analysis. By acknowledging and incorporating the impact of external factors, we can anticipate a more convoluted and realistic representation of Amazon stock prices.

References:

[1] “Amazon.com, Inc. (AMZN) Stock Price, News, Quote & History - Yahoo Finance.” Accessed: Dec. 07, 2023. [Online]. Available: <https://finance.yahoo.com/quote/AMZN/>

**Appendix**:

**1.1.1  
Regression Equation**

|  |  |  |
| --- | --- | --- |
| log(Adj Close) | = | 0.8277 + 0.024427 index |

**Model Summary**

|  |  |  |  |
| --- | --- | --- | --- |
| S | R-sq | R-sq(adj) | R-sq(pred) |
| 0.182725 | 97.99% | 97.98% | 97.94% |

**Analysis of Variance**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Source | DF | Adj SS | Adj MS | F-Value | P-Value |
| Regression | 1 | 289.988 | 289.988 | 8685.27 | 0.000 |
| index | 1 | 289.988 | 289.988 | 8685.27 | 0.000 |
| Error | 178 | 5.943 | 0.033 |  |  |
| Total | 179 | 295.931 |  |  |  |

**Durbin-Watson Statistic**

|  |  |
| --- | --- |
| Durbin-Watson Statistic = | 0.260770 |

**1.1.2  
Regression Equation**

|  |  |  |
| --- | --- | --- |
| log(Adj Close) | = | 0.1378 + 0.003205 index + 0.8664 log(Adj Close)\_Lag1 |

**Model Summary**

|  |  |  |  |
| --- | --- | --- | --- |
| S | R-sq | R-sq(adj) | R-sq(pred) |
| 0.0905172 | 99.50% | 99.50% | 99.48% |

**Analysis of Variance**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Source | DF | Adj SS | Adj MS | F-Value | P-Value |
| Regression | 2 | 288.672 | 144.336 | 17616.21 | 0.000 |
| index | 1 | 0.100 | 0.100 | 12.19 | 0.001 |
| log(Adj Close)\_Lag1 | 1 | 4.452 | 4.452 | 543.38 | 0.000 |
| Error | 176 | 1.442 | 0.008 |  |  |
| Total | 178 | 290.114 |  |  |  |

**Durbin-Watson Statistic**

|  |  |
| --- | --- |
| Durbin-Watson Statistic = | 1.721 |

A graph with a line

Description automatically generated

**1.1.3**

**Smoothing Constants**

|  |  |
| --- | --- |
| α (level) | 0.8 |
| γ (trend) | 0.1 |

**Accuracy Measures**

|  |  |
| --- | --- |
| MAPE | 3.72949 |
| MAD | 0.07483 |
| MSD | 0.01018 |

A graph with red and blue lines

Description automatically generated

**1.1.4**

Smoothing Constants

|  |  |
| --- | --- |
| α (level) | 0.9 |
| γ (trend) | 0.2 |

Accuracy Measures

|  |  |
| --- | --- |
| MAPE | 3.86330 |
| MAD | 0.07768 |
| MSD | 0.01026 |

A graph with red and blue lines

Description automatically generated

**1.1.5**Final Estimates of Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type** | **Coef** | **SE Coef** | **T-Value** | **P-Value** |
| AR   1 | -0.1485 | 0.0773 | -1.92 | 0.056 |
| AR   2 | -0.0169 | 0.0783 | -0.22 | 0.829 |
| AR   3 | 0.0333 | 0.0775 | 0.43 | 0.669 |
| MA   1 | 0.986964 | 0.000313 | 3149.17 | 0.000 |
| Constant | 0.0129 | 0.0103 | 1.25 | 0.215 |

Residual Sums of Squares

|  |  |  |
| --- | --- | --- |
| **DF** | **SS** | **MS** |
| 173 | 4317.73 | 24.9580 |

Back forecasts excluded

Modified Box-Pierce (Ljung-Box) Chi-Square Statistic

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lag** | **12** | **24** | **36** | **48** |
| Chi-Square | 20.16 | 55.99 | 69.13 | 75.49 |
| DF | 7 | 19 | 31 | 43 |
| P-Value | 0.005 | 0.000 | 0.000 | 0.002 |

**1.1.6**

**Final Estimates of Parameters**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type** | **Coef** | **SE Coef** | **T-Value** | **P-Value** |
| AR   1 | -1.853 | 0.195 | -9.50 | 0.000 |
| AR   2 | -1.192 | 0.388 | -3.07 | 0.003 |
| AR   3 | -0.095 | 0.269 | -0.35 | 0.723 |
| MA   1 | -0.942 | 0.161 | -5.83 | 0.000 |
| MA   2 | 0.529 | 0.189 | 2.80 | 0.006 |
| MA   3 | 1.1061 | 0.0860 | 12.87 | 0.000 |
| MA   4 | 0.225 | 0.241 | 0.93 | 0.352 |

Differencing: 2 Regular  
Number of observations after differencing: 178

**Model Summary**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **DF** | **SS** | **MS** | **MSD** | **AICc** | **AIC** | **BIC** |
| 171 | 4034.95 | 23.5962 | 22.6682 | 1077.51 | 1076.66 | 1102.11 |

MS = variance of the white noise series

**Modified Box-Pierce (Ljung-Box) Chi-Square Statistic**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lag** | **12** | **24** | **36** | **48** |
| Chi-Square | 27.19 | 47.36 | 63.67 | 68.67 |
| DF | 5 | 17 | 29 | 41 |
| P-Value | 0.000 | 0.000 | 0.000 | 0.004 |

**1.1.7**

**Final Estimates of Parameters**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type | Coef | SE Coef | T-Value | P-Value |
| AR   1 | -0.1349 | 0.0760 | -1.77 | 0.078 |
| MA   1 | 0.98084 | 0.00211 | 465.31 | 0.000 |

Differencing: 2 Regular  
Number of observations after differencing: 178

**Model Summary**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| DF | SS | MS | MSD | AICc | AIC | BIC |
| 176 | 4364.18 | 24.7965 | 24.5179 | 1084.31 | 1084.17 | 1093.72 |

MS = variance of the white noise series

**Modified Box-Pierce (Ljung-Box) Chi-Square Statistic**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lag | 12 | 24 | 36 | 48 |
| Chi-Square | 20.53 | 53.88 | 67.41 | 73.47 |
| DF | 10 | 22 | 34 | 46 |
| P-Value | 0.025 | 0.000 | 0.001 | 0.006 |

**1.1.8**

import pandas as pd

from statsmodels.tsa.holtwinters import ExponentialSmoothing

from sklearn.metrics import mean\_absolute\_error

import numpy as np

import matplotlib.pyplot as plt

from mpl\_toolkits.mplot3d import Axes3D

# Read data from Excel

excel\_path = 'log.xlsx'

df = pd.read\_excel(excel\_path)

ts\_data = df['log\_values'] # Replace 'your\_column\_name' with the actual column name

# Define alpha and beta values

alpha\_values = np.arange(0.1, 1.0, 0.1)

beta\_values = np.arange(0.1, 1.0, 0.1)

# Initialize variables to store the best model and MAPE

best\_alpha = None

best\_beta = None

best\_mape = float('inf')

# Lists to store results for plotting

alpha\_list, beta\_list, mape\_list = [], [], []

# Iterate over alpha and beta values

for alpha in alpha\_values:

for beta in beta\_values:

# Fit double exponential smoothing model

model = ExponentialSmoothing(ts\_data,seasonal=None, initialization\_method="estimated")

fit = model.fit(smoothing\_level=alpha, smoothing\_trend=beta)

# Make predictions on train data

predictions = fit.fittedvalues

# Calculate MAPE on train data

mape = mean\_absolute\_error(ts\_data, predictions) / np.mean(ts\_data) \* 100

# Update best model if current MAPE is lower

if mape < best\_mape:

best\_alpha = alpha

best\_beta = beta

best\_mape = mape

# Append results for plotting

alpha\_list.append(alpha)

beta\_list.append(beta)

mape\_list.append(mape)

# Plot 3D surface plot of MAPE values

fig = plt.figure()

ax = fig.add\_subplot(111, projection='3d')

ax.scatter(alpha\_list, beta\_list, mape\_list, c='r', marker='o')

ax.set\_xlabel('Alpha')

ax.set\_ylabel('Beta')

ax.set\_zlabel('MAPE')

ax.set\_title('MAPE for Different Alpha and Beta Values')

# Print the best alpha, beta, and corresponding MAPE

print(f"Best alpha: {best\_alpha}")

print(f"Best beta: {best\_beta}")

plt.show()

**1.1.9**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from statsmodels.tsa.holtwinters import ExponentialSmoothing

# Read the Excel file

amzn = pd.read\_excel("amzn.xlsx")

ts\_data = pd.Series(amzn['value'].values)

num\_boot = 1000

num\_ahead = 12

alpha = 0.9

beta = 0.1

# Generate bootstrap samples and forecasts

forecasts = pd.DataFrame()

for i in range(num\_boot):

# Fit Exponential Smoothing model to bootstrapped data

new\_obs = ts\_data + np.random.choice(ts\_data, len(ts\_data), replace=True)

ets\_model = ExponentialSmoothing(new\_obs, trend='add', seasonal='add', seasonal\_periods=12)

result\_ets = ets\_model.fit(smoothing\_level=alpha, smoothing\_trend=beta)

# Forecast using the bootstrap model

forecast = result\_ets.forecast(steps=num\_ahead)

forecast\_mean = forecast

# Store forecasts for all bootstrap models

forecasts[f'Boot\_{i + 1}'] = forecast\_mean

# Calculate forecast bounds for each index

forecast\_bounds = pd.DataFrame({

'2.5th Percentile': forecasts.quantile(0.025, axis=1),

'97.5th Percentile': forecasts.quantile(0.975, axis=1)

})

# Print the results for each index

print("Forecast Bounds:\n", forecast\_bounds)

# Plot the forecasts and prediction limits

plt.plot(ts\_data.index, ts\_data.values, label='Original Time Series')