Data Analytics Report and Executive Summary

Gooden, Nina S. | Student ID #: 009823504

Western Governors University

Dr. William Sewell

December 22, 2022

Abstract

This data analytics graduate capstone report will explore the sales data for Montgomery County of Maryland, USA. In doing so, this analysis will create a time series model which will predict the alcohol sales in a forecast that expands outside of the available data set. This forecast will be valuable for continued predictions of under, over, and par operating (Sai Ram, 2020) for the county. In addition to this forecast, the researcher will provide visualizations to support insights derived from the data, as well as sturdy, reproduceable, and scalable environments. Limitations and potential courses of action will also be examined in these data reports

Keywords: Time Series Analysis, ARIMA analysis, forecast, sales prediction, master's degree, degree capstone

Data Analytics Report and Executive Summary

The contribution of this study to the field of Data Analytics and the MSDA program is to create a time series model which will predict the alcohol sales for Montgomery County of Maryland, USA. By creating this forecast, the county will be able to compare their real sales against the predicted thresholds, thus allowing it to measure whether it is under-, over-, or par operating (Sai Ram, 2020). A time series analysis analyzes a sequence of data points collected over an interval of time in order to measure how variables change over time. (Time Series Analysis: Definition, Types, Techniques, and When It’s Used, n.d.) This study will utilize autoregressive integrated moving average (ARIMA) to describe autocorrelations in the data, while also using exploratory data techniques to derive general insights about the dataset. Lastly, the researchers will package findings in a digital dashboard for general consumption. It is the researcher’s hypothesis that there will be a clear prediction derived from this dataset.

# Research Question

The standing research question for this analysis is as follows: can an ARIMA analysis predict sales of each item type for Montgomery County for the following quarter?

## Hypothesis

The null hypothesis of this evaluation is that the time series model cannot be made from the available dataset.

The alternative hypothesis of this evaluation is that the time series model can be made from the available dataset.

The project will seek to create a forecast model for the number of sales in the next quarter for the combined efforts of warehouse and retail distribution points for Montgomery County of Maryland. Support for the alternative hypothesis is found in Time series analysis: validating effect of changes with the statement “time series hypothesis testing talks about how we identify whether different time periods have significantly different observation.” (Tunggawan, 2018)

# Data Governance

## Data Analysis Goals and Objective

The primary goal of this analysis is to identify patterns in earned revenue to support the medical center’s evaluation of their readmission impact. Researchers will be testing several versions of the available ARIMA time series model creations to find one that is best suited for this analysis. A time series analysis analyzes a sequence of data points collected over an interval of time to measure how variables change over time. (Time Series Analysis: Definition, Types, Techniques, and When It’s Used, n.d.) This study will utilize autoregressive integrated moving average (ARIMA) to describe autocorrelations in the data, while also using exploratory data techniques to derive general insights about the dataset. Lastly, the researchers will package findings in a digital dashboard for general consumption. It is the researcher’s hypothesis that there will be a clear prediction derived from this dataset.

### **The data**

The data needed to be collected for the question is the publicly available information provided by the Montgomery Country of Maryland local government ESB Service *Warehouse and Retail Sales - Catalog.* (n.d.). The data was last updated December 9, 2022 and is updated monthly. There are 307,646 records in the data set and 9 columns.

#### Data Structure***.***

The data set is made available through the data.gov website. The data set includes the following variables of year, month, supplier, item code, item description, item type, retail sales, retail transfers, and warehouse sales. The predictor variables are broken down in (Figure 1).

#### Data Accessibility***.***

Montgomery County of Maryland has made this information publicly available through the data.gov website. It is intended for public access and use, and no license information was provided. The dataset is updated regularly and has been consistently managed since 2017, meaning the data available is healthy and of sufficient size. There will be no information in the final evaluation that would make the companies listed in these transactions identifiable.

##### Data Limitations

The information available through public forums does not include price data or revenue information. As such, the time series evaluation will need to focus on the number of sales, rather than the value of said transactions.

##### Data Delimitations

Data supplies names for companies that are found in this dataset. Removing said names will bring another level of security to the report, which is considered best practices in data management (Zaugg, 2018).

### **Data Gathering**

Data will be downloaded from a publicly available CSV file from the data.gov website which shows data for all warehouse and retail sales from Montgomery Country of Maryland. Any entries that have missing inputs for retail or warehouse sales will be replaced by standard deviation or mean data. Missing data in the ITEM TYPE field will be manually evaluated and inputted, which is the ideal substitution method for easily located, low-cost data (Mesidor, 2021). Records missing date information will be removed, as time series models work with the complete data and require input of the missing values prior to modeling. (Kumar, 2022). No other columns will be evaluated in this analysis. The data quality is very high, as it has been collected by local government and maintained for five years. Data will be made stationary if it is not initially and correlation and auto correlation will be plotted visually. Overall data sparsity is < 5%.

## Data Analytics Tools and Techniques

A succinct overview of the steps proposed to accomplish and measure this modeling goal has been developed thusly:

1. Data will be checked for any imperfections and normalized as necessary.
2. Unnecessary columns will be removed from the analysis.
3. Exploratory visualizations will be created for cleaned data to gather insights on which item type contributes what percentage of sales.
4. Initial visualization will be created using pyplot.
5. Evaluate stationarity and correct if necessary, using autocorrelation function (ACF) and partial autocorrelation function (PACF) to find the p and q values for ARIMA. (Zvornicanin, 2022)
6. Portion the data, so that a segment may be held over for testing and assessment of the model.
7. ARIMA model construction and fitting.
8. Model accuracy measured via withheld data, as well as mean square root evaluation.
9. Initial visualizations will be created.
10. Data will be exported in order to create dashboard visualizations in Tableau.

The auto\_arima and ARIMA functions will be used to create forecast data, though it is important to note that ARIMA has difficult prediction turning points. (Understanding ARIMA Models for Machine Learning. (n.d.)). The available data ends in 2020, which was a mark of a global turning point and thus the prediction model will likely exceed the reality of data.

#### Justification of Tools and Techniques***.***

Python will be used for the duration of the project. According to (Manokhin, 2022), “when it comes to forecasting and time series in 2022, Python is the no-brainer choice.” In addition, (Alam, 2021) states that ARIMA is “arguably the most popular and widely used statistical technique for forecasting.” These sentiments justify a strong foundation for time series analysis that is accurate, reproducible, and scalable.

# Data Preparation

## Environment construction

For research to be duplicatable, efforts must be made to ensure the tools and settings utilized fit with community standards. To align with these efforts, the analysis began in Jupyter Notebook, via the text editing capabilities of Visual Studio Code. Packages and libraries were imported into the environment as follows:

1. pandas as pd, used for DataFrame components and data manipulation
2. numpy as np, utilized for array work and base Python interactions
3. matplotlib.pyplot as plt, and seaborn as sns used for data visualizations
4. sklearn used for complex mathematical components, as well as testing model accuracy
5. statsmodels.api as sm, used as the primary interface for the ARIMA modeling and data preparation
6. datetime, use for data preparation

## Data Management

### **Cleaning**

After importing the dataset, exploratory functions were launched in order to investigate the data further. Columns unnecessary for the evaluation were dropped.

A cursory evaluation of the data conducted before the implementation of code incorrectly assessed that there was no missing data in the dataset. As such, it was necessary to correct this lack of information before the ARIMA model could be created.

The .fillna() method was used for a single manual input from a missing value in the ‘ITEM TYPE’ column. This same method was also used later to fill values in the numerical columns ‘RETAIL SALES’ and ‘WAREHOUSE SALES’, with the mean column values.

One of the key characteristics of a time series analysis is the use of dates as index markers for growth and attrition. Later in the evaluation, the researchers realized there were date values missing. As such, in future passes of this evaluation, the data was prepared by creating a list of the missing values and using .append() to add them to ‘MONTH’ and ‘YEAR’ columns manually.

After renaming all columns to no longer be in caps case, the researchers generated a ‘total\_sales’ column from the ‘retail\_sales’ and ‘warehouse\_sales’ columns. This was necessary to create a single target variable for the later analysis.

In order to continue the conversion of the ‘month’ and ‘year’ variables to the necessary datetime object for ARIMA, a ‘day’ column was created with a default value of 1. After which, the to\_datetime() function was utilized to create a single, date column for the time series analysis. The DataFrame was then sorted by the new ‘date’ column.

The resulting file was exported for later use, as it would be heavily manipulated for exploratory visualization.

Columns deemed unnecessary were dropped. Visualizations for the product types available in the dataset were created to compare the contribution of each product. Figure 2 notes that there was a considerable cost for certain products, and so Figure 3 was created without those items. Beer had a disproportionately high contribution to earnings. Due to this observation, researchers decided that there was no additional analysis needed to identify which “type” of sale would be evaluated by the model. Removing Beer in any iteration would result in too few data points.

The exported data was uploaded again and assessed for continuity. Different columns were dropped, leaving only the sales data and the dates. The date data was set at the index for the DataFrame and entries that shared a date were combined with .sum().round(2) and organized in ascending order.

The research noted that once all the duplicates were removed, there were only 43 entries left in the DataFrame, found in Figure 4. This meant that the model would be more sensitive to each fluctuation and would result in large variances.

### **Stationarity**

A visualization of the initial dataset—properly cleaned—was created, found in Figure 5. As the data was not visually stationary, an Augmented Dickey-Fuller unit root test (.adfuller()) was required. The results, found in Table 1, confirmed that the null hypothesis was not rejectable based on the current data.

To transform the dataset to stationary researchers used differencing to subtract each point from the previous value in the primary column. Differencing is a popular and widely used data transform for making time series data stationary. (Brownlee, 2020)

After completing this step, the data was visualized again in order to conduct an initial measurement of success. The graphic in Figure 6 is the result of that visualization. After the differencing was complete, a second Augmented Dickey-Fuller unit root test (.adfuller()) was required. The results of that test can be found in Table 2. Researchers observed that the absolute value of the ADF Statistic value was greater than the critical value. In addition, the p-Value was below 0.05, confirming that the data was now stationary and the null hypothesis was rejectable. (Chaudhary, 2021)

### **Identifying AR and MA parameters**

In order to run the autocorrelation analysis (acf()) and the partial autocorrelation analysis (pacf()), researchers used the train\_test\_split function on the differenced data to split the data into two groups. The researchers opted to use the recommended 4:1 ratio for said split.

The acf() function was applied to the train dataset in order to correlate the points of the dataset with a lagged version of itself. The autocorrelation function begins at lag 0 and results in a correlation of 1. Figure 7 is a visualization of the resulting graph.

The pacf() function was then applied to the train dataset in order to correlate the points of the dataset by each successive lagged term. The correlation between observed points is considered as correlated to observations at other time junctions. Figure 8 is a visualization of the resulting graph.

Finally, the data was exported for posterity.

# Model Identification and Analysis

## Pre-modeling Assessment

### **Initial Visualizations**

Two visualizations were created to measure the potential performance of the model before the model was created. The seasonal decomposition is a useful and quick tool created to look at the trend, seasonality, and noise present in the dataset. This visualization can be found as Figure 9. The researchers paid close attention to the Seasonality and Residuals lines especially. The resulting flat visualization was what was expected.

The second visualization created to review the state of the data was the spectral density analysis. This visual, found in Figure 10 is a “frequency domain representation of a time series that is directly related to the autocovariance time domain representation.” (*12.1 Estimating the Spectral Density | STAT 510*, n.d.). This analysis helps to determine the frequency of the sales variables. This graph further exhibits no evidence of seasonal trends.

### **Model Creation**

The auto\_arima function was used to test different arguments for the potential ARIMA model quickly. This was done by performing a stepwise search for the minimum Akaike's Information Criterion (AIC) possible. The returned best model values were ARIMA(2,0,0)(0,0,0)[0] intercept.

A second, independent ARIMA model was created with the given order parameters in order to confirm the assertion from auto\_arima. When run, the SARIMAX Results were within acceptable ranges, with p\_Values that were not zero, but largely less than .05. Model and results found in Figure 11.

With the model created, the data was split again, and the updated test and train models were filtered into the .predict() function in order to produce prediction information on the train and test values. This is documented in Figure 12. A visualization of the resulting prediction can be found as Figure 13.

### **Model Test**

An analysis of the .mean() values of the test values returned a relatively large number. Additionally, the root mean square error—the standard deviation of the spread of residuals—was comparatively stable.

### **Research Question Prediction**

The research question was whether or not a time series model could be used to predict the next quarter’s sales data for each item type in the dataset. While there was not enough support to itemize the prediction, a time series model was created for overall sales. Figure 13 offers insights as to the code used in the creation of that prediction.

# Data Summary and Implications

Though the overall trend in the data is steadily increasing, there is a considerable downturn approaching, according to the model **(Pierre, 2021).** The implication of this downward trend is that the county will suffer from sales loss at the beginning of the year. A sound recommendation is that the county spend time leading up to this point to put in place processes and spending limits to lessen the impact of the revenue loss.

In order to better analysis data in the future, it would be beneficial for the county to record data on sales per day, rather than per month. It would also be beneficial for the county to record gains and losses for the missing dates so that the standard deviation information could be replaced with reality. It is the researcher’s recommendation that this evaluation be run on a quarterly basis to continue to reap the benefits of forecasting.

References

Time Series Analysis: Definition, Types, Techniques, and When It’s Used. (n.d.). Tableau. https://www.tableau.com/learn/articles/time-series-analysis

Sai Ram, K. (2020, February 28). *Time Series Analysis and Forecasting: Novel Business Perspectives*. AltexSoft. <https://www.altexsoft.com/blog/business/time-series-analysis-and-forecasting-novel-business-perspectives/>

*Warehouse and Retail Sales - Catalog*. (n.d.). <https://catalog.data.gov/dataset/warehouse> and-retail-sales

Kumar, S. (2022, April 30). *4 Techniques to Handle Missing values in Time Series Data*. Medium. https://towardsdatascience.com/4-techniques-to-handle-missing-values-in-time-series-data-c3568589b5a8

*Understanding ARIMA Models for Machine Learning*. (n.d.). Capital One. https://www.capitalone.com/tech/machine-learning/understanding-arima-models/

Alam, M. (2021, December 15). Comparing the performance of forecasting models: Holt-Winters vs ARIMA. Medium. <https://towardsdatascience.com/comparing-the-performance-of-forecasting-models-holt-winters-vs-arima-e226af99205f>

Zvornicanin, E. (2022, November 8). Choosing the best q and p from ACF and PACF plots in ARMA-type modeling. Baeldung. <https://www.baeldung.com/cs/acf-pacf-plots-arma-modeling>

Manokhin, V., PhD. (2022, December 16). Python vs R for time-series forecasting - Valery Manokhin, PhD, MBA, CQF. Medium. <https://valeman.medium.com/python-vs-r-for-time-series-forecasting-395390432598>

Zaugg, C. (2018, April 23). Data Redaction: What it is, why you need it, and how to start. Reveal. <https://opin.com/intro-to-redaction/>

Mesidor, F. (2021, December 11). A basic guide to time series analysis - Towards Data Science. Medium. <https://towardsdatascience.com/a-basic-guide-into-time-series-analysis-2ad1979c7438>

Tunggawan, E. (2018, October 27). Time series analysis: validating effect of changes. Elvyna Tunggawan. <https://elvyna.github.io/2018/time-series-hypothesis-testing/>

Brownlee, J. (2020, June 23). *How to Remove Trends and Seasonality with a Difference Transform in Python*. Machine Learning Mastery. https://machinelearningmastery.com/remove-trends-seasonality-difference-transform-python/

Chaudhary, M. (2021, December 14). *Why is Augmented Dickey–Fuller test (ADF Test) so important in Time Series Analysis*. Medium. <https://medium.com/@cmukesh8688/why-is-augmented-dickey-fuller-test-adf-test-so-important-in-time-series-analysis-6fc97c6be2f0>

*12.1 Estimating the Spectral Density | STAT 510. (n.d.)*. PennState: Statistics Online Courses. <https://online.stat.psu.edu/stat510/lesson/12/12.1>

Pierre, S. (2021, July 16). A Guide to Time Series Analysis in Python | Built In. A Guide to Time Series Analysis in Python | Built In. https://builtin.com/data-science/time-series-python

Tables

Table 1

Initial Augmented Dickey-Fuller unit root test

| df\_test = ts.adfuller(df[‘total\_sales’] | |
| --- | --- |
| ADF Statistic | -1.706677 |
| p-Value | 0.427676 |
| Lags | 1.000000 |
| Observations | 41.000000 |
| Critical Values (1%) | -3.600983 |
| Critical Values (5%) | -2.935135 |
| Critical Values (10%) | -2.605963 |

Note: A statistical significance test that aids in the testing of the null and alternative hypothesis. Results in the creation of test statistics, as well as p-Values which can be utilized to inference as to whether a given dataset is stationary or not.

Table 2

Augmented Dickey-Fuller unit root test post-differencing

| df\_test = ts.adfuller(df[‘total\_sales’] | |
| --- | --- |
| ADF Statistic | -4.594161 |
| p-Value | 0.000132 |
| Lags | 2.000000 |
| Observations | 39.000000 |
| Critical Values (1%) | -3.610400 |
| Critical Values (5%) | -2.939109 |
| Critical Values (10%) | -2.608063 |

Note: A statistical significance test that aids in the testing of the null and alternative hypothesis. Results in the creation of test statistics, as well as p-Values which can be utilized to inference as to whether a given dataset is stationary or not.

Table 3

Augmented Dickey-Fuller unit root test post-differencing

| stepwise\_fit = auto\_arima(df['total\_sales'], trace = True, suppress\_warnings = True) |
| --- |
| Performing stepwise search to minimize aic |
| ARIMA(2,0,2)(0,0,0)[0] intercept : AIC=1091.016, Time=0.05 sec |
| ARIMA(0,0,0)(0,0,0)[0] intercept : AIC=1111.779, Time=0.01 sec |
| ARIMA(1,0,0)(0,0,0)[0] intercept : AIC=1094.260, Time=0.02 sec |
| ARIMA(0,0,1)(0,0,0)[0] intercept : AIC=1105.847, Time=0.02 sec |
| ARIMA(0,0,0)(0,0,0)[0] : AIC=1231.374, Time=0.01 sec |
| ARIMA(1,0,2)(0,0,0)[0] intercept : AIC=1091.692, Time=0.03 sec |
| ARIMA(2,0,1)(0,0,0)[0] intercept : AIC=1088.974, Time=0.04 sec |
| ARIMA(1,0,1)(0,0,0)[0] intercept : AIC=1093.148, Time=0.03 sec |
| ARIMA(2,0,0)(0,0,0)[0] intercept : AIC=1088.892, Time=0.03 sec |
| ARIMA(3,0,0)(0,0,0)[0] intercept : AIC=1088.994, Time=0.04 sec |
| ARIMA(3,0,1)(0,0,0)[0] intercept : AIC=1090.845, Time=0.05 sec |
| ARIMA(2,0,0)(0,0,0)[0] : AIC=inf, Time=0.02 sec |
| Best model: ARIMA(2,0,0)(0,0,0)[0] intercept |
| Total fit time: 0.360 seconds |

Note: Used to test multiple parameters for an ARIMA model, in order to find the lowest Akaike’s Information Criterion (AIC) possible. The resulting best model returns the ideal order parameters for ARIMA, as well as a second set of information that measures seasonality and trends.

Figures

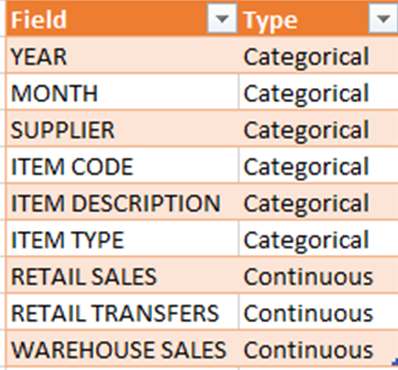


Figure 1. organized list of the available columns and the corresponding types for the original dataset.

Chart, waterfall chart

Description automatically generated

Figure 2. Plotted in seaborn as a visualization of all products. Visualization is disturbed by the large number of negative value items in the dataset.

Chart, bar chart

Description automatically generated Figure 3. Adjusted product visualization removes negative-performing products to evaluate those that contribute to total sales.

Text

Description automatically generated

Figure 4. Printed info() before and after grouping to show that there were 307,663 records before combination and only 43 after.

Chart, histogram

Description automatically generatedFigure 5. Visualization of DataFrame after all variables have been cleaned and prepared.

Chart, line chart, histogram

Description automatically generatedFigure 6. Visualization of DataFrame after differencing has been implemented.

Chart, line chart

Description automatically generatedFigure 7. Visualization of ACF function on training data.

Chart, line chart

Description automatically generatedFigure 8. Visualization of ACF function on training data, black line added to mark 0.05 as p-Value.

A picture containing line chart

Description automatically generated

Figure 9. Visualization of the seasonal\_decompose() function. The resulting object contains original, trend, seasonality, and residuals data for researchers to evaluate.

Chart, line chart

Description automatically generated

Figure 10. Power spectral density visual, created to mark power, strength, and frequency of data.

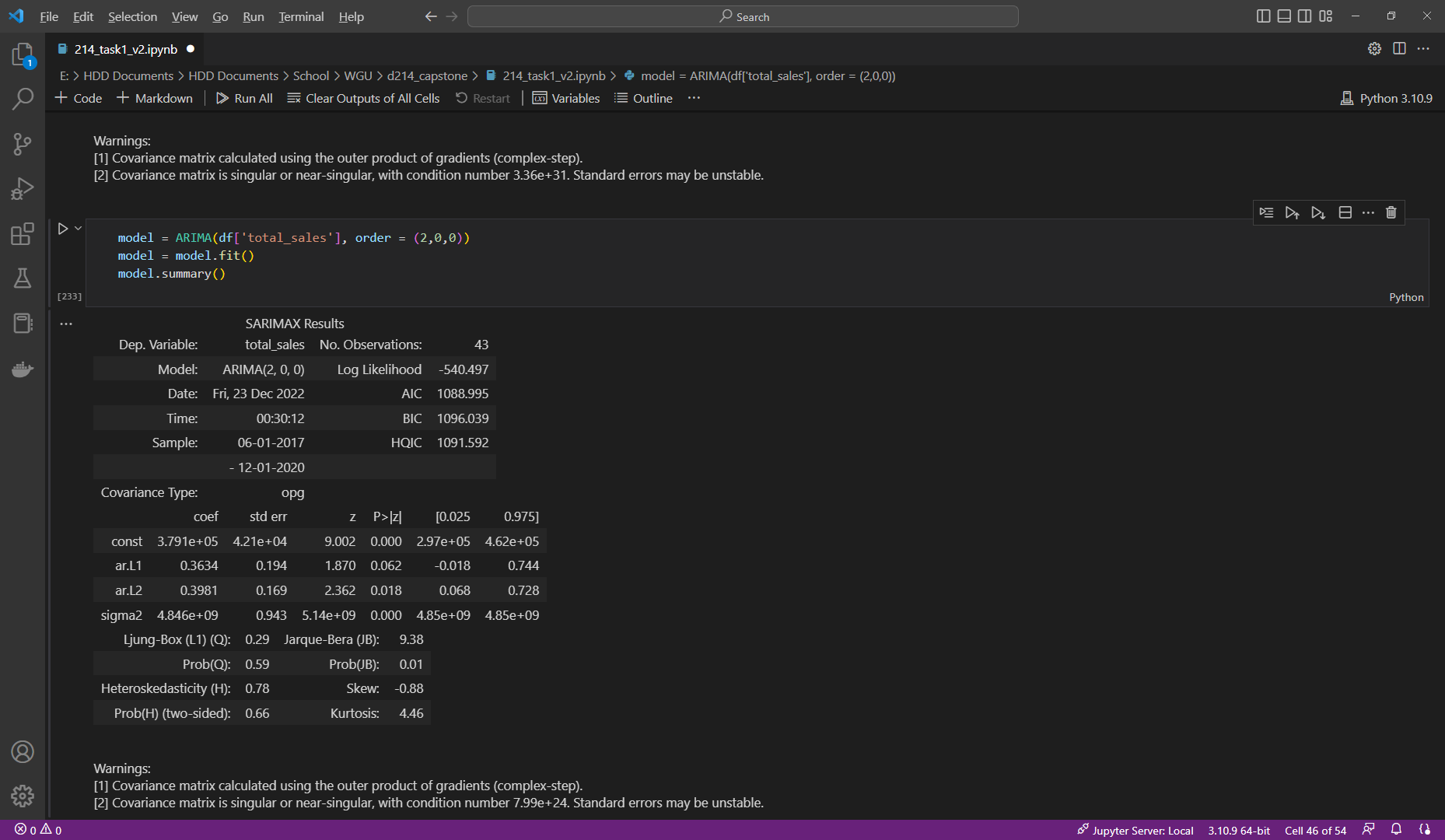


Figure 11. SARIMAX Results from independent model creation using defined parameters from auto\_arima.

A screenshot of a computer

Description automatically generated with medium confidence

*Figure 12.* Test and train variables updated for predictions.

Chart, line chart

Description automatically generated

*Figure 13.* Visualization of model prediction, compared to test data for total sales.

Graphical user interface, text

Description automatically generated

*Figure 14.* Code used to generate the prediction model, fit to extend for 3 months after the dataset concludes.

Chart, line chart

Description automatically generated

*Figure 14.* Model fitted to training data—organized by month.