Shawn Wheeler

D213 Task I

000364855

Table of Contents

[Part I: Research Question 3](#_Toc170493635)

[A1: Research Question 3](#_Toc170493636)

[A2: Objectives or Goals 3](#_Toc170493637)

[Part II: Method Justification 3](#_Toc170493638)

[B: Summary of Assumptions 3](#_Toc170493639)

[Part III: Data Preparation 3](#_Toc170493640)

[C1: Line Graph Visualization 3](#_Toc170493641)

[C2: Time Step Formatting 4](#_Toc170493642)

[C3: Stationarity 5](#_Toc170493643)

[C4: Steps to Prepare the Data 6](#_Toc170493644)

[C5: Prepared Data Set 8](#_Toc170493645)

[Part IV: Model Identification and Analysis 9](#_Toc170493646)

[D1: Report Findings and Visualizations 9](#_Toc170493647)

[D2: Arima Model 13](#_Toc170493648)

[D3: Forecasting Using Arima Model 14](#_Toc170493649)

[D4: Output and Calculations 15](#_Toc170493650)

[D5: Code 22](#_Toc170493651)

[Part V: Data Summary and Implications 22](#_Toc170493652)

[E1: Results 22](#_Toc170493653)

[E2: Annotated Visualization 23](#_Toc170493654)

[E3: Recommendation 23](#_Toc170493655)

[Part VI: Reporting 24](#_Toc170493656)

[F: Reporting 24](#_Toc170493657)

[G: Sources for Third-Party Code 24](#_Toc170493658)

[H: Sources 24](#_Toc170493659)

## Part I: Research Question

### A1: Research Question

Is it possible to use the time-series dataset in order to predict six months of future revenue using ARIMA?

### A2: Objectives or Goals

The objective of this project is to attempt to forecast six months of revenue in the future. In order to do this, the past two years of revenue data will be analyzed. That data will be used to train an ARIMA model which will then be used to forecast revenue beyond the current date range of the data.

## Part II: Method Justification

### B: Summary of Assumptions

There are assumptions of time series models, such as ARIMA. Firstly, data will exhibit stationarity. Stationarity indicates that the data will have constant distribution over time with a lack of trends. This includes properties like mean, variance, and autocorrelation. Shifting the time will not alter the shape of distribution. (Stephanie, 2016).

One other assumption of time series models is that data points will not be highly correlated with past observations. Using autocorrelation, the correlation between a time series and its own lagged version are measured over time intervals. (GeeksforGeeks, 2020).

Residuals are both normal and independent. Another assumption is that outliers and other anomalous data issues are addressed. (Overload, 2023).

## Part III: Data Preparation

### C1: Line Graph Visualization

*Line Graph of Original Data:*

A graph with blue lines

Description automatically generated

*Code:*

A close-up of a computer code

Description automatically generated

### C2: Time Step Formatting

For this project, there were 730 days worth of revenue data included in the given time series data set. In the original file, they are listed as only days 1-731. There were no measurement gaps detected in the data set. In order to convert these days into their respective dates, the column for Day was modified using to\_timedelta from pandas and a new column, Date, was created to replace the original column. Therefore, the data set only contains the Revenue column and the Date column which begins on 2022-01-01 and ends on 2024-01-01 as opposed to simply a number of days. Finally, the Date column was set as the index.

*Code:*

A screenshot of a computer code

Description automatically generated

### C3: Stationarity

The stationarity of the time series data set with no modifications was tested. The data was graphed to examine stationarity and detect trends. As shown below, there are no obvious trends within the data set. A Dickey-Fuller test was also performed and p-value extracted. A p-value of less than .05 indicates stationarity. Below, the p-value is shown to be 0.2 for the original data set; therefore, it is not stationary.

*Screenshot of Original Data Trendline:*

*A graph with blue lines and orange dots

Description automatically generated*

*Screenshot of Rolling Mean and Standard Deviation:*

*A graph of a graph showing the value of a stock market

Description automatically generated with medium confidence*

*Screenshot of Dickey Fuller results:*

**

*Code:*

A screenshot of a computer program

Description automatically generated

### C4: Steps to Prepare the Data

1. Import packages and libraries

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from pmdarima import auto\_arima

from statsmodels.tsa.arima.model import ARIMA

from statsmodels.tsa.seasonal import seasonal\_decompose

from statsmodels.graphics.tsaplots import plot\_acf, plot\_pacf

from statsmodels.tsa.stattools import adfuller

from sklearn.model\_selection import train\_test\_split

%matplotlib inline

import warnings

warnings.filterwarnings('ignore')

1. Open .csv for analysis

df = pd.read\_csv("C:/Users/Owner/medical\_time\_series.csv")

1. Data Exploration

df.info()

df.head()

1. Checking for duplicates and missing data

df.duplicated().value\_counts()

df.isnull().sum()

1. Timestep formatting and set index to Date column

# Date

initial\_date = pd.to\_datetime('2022-01-01')

# Fix Day column

df['Date'] = pd.to\_timedelta(df['Day'] - 1, unit='D') + initial\_date

df.drop('Day', axis=1, inplace=True)

# Set index

df.set\_index('Date', inplace=True)

1. Test for stationarity by visually examining data and use Dickey-Fuller

# Line graph

df.plot(figsize=(10,5), ylabel='Revenue')

plt.title('Original Data Line Graph')

plt.legend()

# Graph with trend line (Zach, 2022)

df.plot(figsize=(10,5), ylabel='Revenue')

# Define

trend\_x = np.arange(len(df.index))

trend\_y = df['Revenue']

# Calculate

trend\_z = np.polyfit(trend\_x, trend\_y, 2)

trend\_p = np.poly1d(trend\_z)

# Change to dates for graph

trend\_x\_dates = df.index.date

# Plot trendline

plt.plot(trend\_x\_dates, trend\_p(trend\_x), "--", label='Trend Line')

plt.legend()

plt.title('Original Data with Trend Line')

# Graph revenue data

monthly\_rolling\_mean = df.rolling(window=30).mean()

monthly\_rolling\_std = df.rolling(window=30).std()

plt.figure(figsize=[10,5])

plt.plot(df, label = 'Dataset')

plt.plot(monthly\_rolling\_mean, ls='-.', label='Rolling Mean')

plt.plot(monthly\_rolling\_std, ls='--', label='Rolling Standard Deviation')

plt.legend()

plt.xlabel('Date')

plt.ylabel('Revenue (in millions)')

plt.title('Original Data with Rolling Mean and Rolling Standard Deviation')

plt.show()

# Run ADF (Dickey-Fuller) and show ADF

dickeyfuller = adfuller(df)

print('ADF: ', dickeyfuller[0].round(2))

print('p-value: ', dickeyfuller[1].round(2))

1. Difference data and examine

df\_diff = df.diff().dropna()

# Graph

plt.figure(figsize=[10,5])

plt.title('Differenced Dataset')

plt.xlabel('Date')

plt.ylabel('Value')

plt.plot(df\_diff)

plt.show()

1. Testing stationarity of differenced data

# Rerun stationarity tests on differenced data

dickeyfuller\_diff = adfuller(df\_diff)

print('ADF: ', dickeyfuller\_diff[0].round(2))

print('p-value: ', dickeyfuller\_diff[1].round(2))

1. Split Data and examine

train, test = train\_test\_split(df, test\_size=0.2, shuffle=False, random\_state=13)

train

test

1. Save new files

# Saving new files

train.to\_csv('d213task1train.csv')

test.to\_csv('d213task1test.csv')

### C5: Prepared Data Set

*See Attached .csv Files:* d213task1train.csv , d213task1test.csv

## 

## Part IV: Model Identification and Analysis

### D1: Report Findings and Visualizations

*The presence or lack of a seasonal component:*

Seasonal components were searched for among the data set. There does appear to be a minor seasonal component as shown by using an additive seasonal decomposition.

*Screenshot of Seasonality:*

*A graph of a blue line

Description automatically generated with medium confidence*

*Trends:*

No overall trends were observed in the visualized data set. The graphs shown include a polynomial trend line, a rolling mean, and a rolling standard deviation.

*Screenshot of Trend Line:*

*A graph with blue lines and orange lines

Description automatically generated*

*Screenshot including rolling mean and standard deviation:*

*A graph of a graph showing the value of a stock market

Description automatically generated with medium confidence*

*Autocorrelation function:*

The autocorrelation function and the partial autocorrelation function were performed. ACF shows the correlation at different lags and can be used to find the value for ‘p’ which is needed to perform ARIMA. A positive value indicates correlation. PACF is similar to ACF except that it removes effects from previous lags. PACF provides information for the ‘q’ value needed for ARIMA. (Ahmed, 2023).

*Screenshot of ACF:*

*A graph with blue dots and numbers

Description automatically generated*

*Screenshot of PACF:*

*A graph with blue dots and numbers

Description automatically generated*

*Spectral density:*

A type of periodogram, the spectral density graph analyzes data for patterns by smoothing to remove irregular variation. Spectral density and ACF show similar information but via different methods. (Anon, n.d.). The graph does appear to be random meaning it exhibits stationarity.

*Screenshot of spectral density:*

*A graph showing a number of blue lines

Description automatically generated*

*Decomposed time series:*

The decomposed data set is shown in the time series below. Decomposing the data using an additive model allows for showing the four graphs seen below for trends, seasonality, and residuals of the data set.

*Screenshot of decomposed graphs:*

*A graph of a graph

Description automatically generated with medium confidence*

*Confirmation of lack of trends in residuals of the decomposed series:*

The residuals from the decomposed time series below do not show any trends.

*Screenshot of residuals:*

*A graph showing a sound wave

Description automatically generated*

### D2: Arima Model

Multiple methods exist to determine the ‘p’, ’d’, and ‘q’ values necessary for ARIMA. In order to determine those values, ACF, PACF, and differencing can be used. Another method is to manually attempt ARIMA using various values and then comparing the AIC (Akaike Information Criteria) to determine the best model to then use for further analysis. In this case, auto-ARIMA was performed. Auto-ARIMA essentially works stepwise through combinations of ‘p’, ‘d’, and ‘q’ and compares the AIC values returning the combination of values that provided the lowest AIC (which is indicative of a superior model). This method provided the values of (1,1,0)(0,0,0)[0] to be used for this project’s ARIMA model. The first set of values represents the ‘p’, ‘d’, and ‘q’ variables whereas the second set of three values represent seasonality components. The third, single, value shows that there is no period.

*Screenshot of auto-ARIMA:*

*A screenshot of a computer

Description automatically generated*

### D3: Forecasting Using Arima Model

Using the values obtained from auto-ARIMA, the next step was to forecast revenue information. The goal mentioned prior was forecasting 60 days in the future. First the model was used to compare a forecast to the testing data for model validation. The model was then used to forecast revenue for dates beyond the available data set. Graphs were created to better visualize the data and forecasts.

*Screenshot of forecast compared to test data:*

A graph with blue lines

Description automatically generated

*Screenshot of future values:*

A screenshot of a computer

Description automatically generated

Screenshot of future forecast:

A graph of a graph

Description automatically generated with medium confidence

### D4: Output and Calculations

1. Code to make and validate data stationarity

# Stationarizing data

# Difference dataset

df\_diff = df.diff().dropna()

# Graph

plt.figure(figsize=[10,5])

plt.title('Differenced Dataset')

plt.xlabel('Date')

plt.ylabel('Value')

plt.plot(df\_diff)

plt.show()

# Rerun stationarity tests on differenced data

dickeyfuller\_diff = adfuller(df\_diff)

print('ADF: ', dickeyfuller\_diff[0].round(2))

print('p-value: ', dickeyfuller\_diff[1].round(2))

*Differenced dataset visual:*

A graph of blue lines

Description automatically generated

*ADF test:*



1. Code for ACF and PACF

# ACF (q value)

plot\_acf(df\_diff, lags=31, zero=False);

# PACF (p value)

plot\_pacf(df\_diff, lags=31, zero=False);

*ACF:*

A graph with blue dots

Description automatically generated

*PACF:*

A graph with blue dots and numbers

Description automatically generated

1. Code for using ARIMA

# Auto ARIMA on original dataset

a\_arima = auto\_arima(df, trace=True, suppress\_warnings=True, stepwise=True)

a\_arima.summary()

# ARIMA Model with training data

arima\_model = ARIMA(train, order=(1,1,0), freq='D')

arima\_results = arima\_model.fit()

arima\_results.summary()

# Graphs from ARIMA results

arima\_results.plot\_diagnostics(figsize=(10,5));

# MAE

mae = arima\_results.mae

print('MAE: ', mae.round(3))

*Auto ARIMA:*

A screenshot of a computer

Description automatically generated

*Fit model:*

A screenshot of a computer

Description automatically generated

*Plot diagnostics:*

*A collage of graphs and diagrams

Description automatically generated*

*MAE:*

**

1. Model summary

*Dataset ARIMA model:*

A screenshot of a data

Description automatically generated

*Plot diagnostics:*

*A collage of graphs and diagrams

Description automatically generated*

*MAE:*

**

1. Predictions for out-of-model data

arima\_model\_full = ARIMA(df, order=(1,1,0), freq='D')

arima\_results\_full = arima\_model\_full.fit()

arima\_results\_full.summary()

# MAE for original data ARIMA

mae = arima\_results\_full.mae

print('MAE: ', mae.round(3))

# Graphs from ARIMA results

arima\_results\_full.plot\_diagnostics(figsize=(10,5));

# Prediction to future dates 6 months

index\_future\_dates = pd.date\_range(start='2024-01-02', end='2024-06-30')

pred = arima\_results.forecast(steps=181)

pred.index = index\_future\_dates

pred

prediction = arima\_results\_full.get\_prediction(start='2024-01-01', end='2024-06-30')

# Prediction mean

mean\_prediction = prediction.predicted\_mean

# Confidence intervals

confidence\_intervals\_prediction\_2 = prediction.conf\_int(alpha=.5)

lower\_limits\_prediction\_2 = confidence\_intervals\_prediction\_2.iloc[:,0]

upper\_limits\_prediction\_2 = confidence\_intervals\_prediction\_2.iloc[:,1]

confidence\_intervals\_prediction\_3 = prediction.conf\_int(alpha=.05)

lower\_limits\_prediction\_3 = confidence\_intervals\_prediction\_3.iloc[:,0]

upper\_limits\_prediction\_3 = confidence\_intervals\_prediction\_3.iloc[:,1]

# Displaying test/training/forecast/intervals

plt.figure(figsize=(10,5))

plt.plot(train['Revenue'], label='Training Data')

plt.plot(test['Revenue'], label='Testing Data')

plt.plot(mean\_prediction, label='Forecast')

plt.fill\_between(confidence\_intervals\_prediction\_2.index, lower\_limits\_prediction\_2, upper\_limits\_prediction\_2, color='purple', alpha=0.2, label='50% confidence')

plt.fill\_between(confidence\_intervals\_prediction\_3.index, lower\_limits\_prediction\_3, upper\_limits\_prediction\_3, color='gray', alpha=0.2, label='95% confidence')

plt.ylim(-5, 30)

plt.legend(loc='upper left')

plt.title('ARIMA Forecasting for 6 Months')

plt.xlabel('Date')

plt.ylabel('Revenue (in millions)')

plt.show()

*Prediction values:*

*A screenshot of a computer

Description automatically generated*

*ARIMA forecasting for six months:*

*A graph of blue and orange lines

Description automatically generated*

### D5: Code

*See Attached File:* d213task1complete.ipynb

## Part V: Data Summary and Implications

### E1: Results

*Selection of ARIMA model:*

Even though ACF, PACF, and differencing were performed, in order to select values for ‘p’, ‘d’, and ‘q’, auto-ARIMA was used. After examining the AIC values for the stepwise procedure, it revealed that (1,1,0)(0,0,0)[0] was the best model. Therefore, the values (1,1,0) were used to then create and fit an ARIMA model.

*Prediction interval of forecast:*

The prediction interval of the dataset is daily. Each individual day has one revenue prediction.

*Justification of forecast length:*

The forecast length selected was six months. Originally, the data set contained two years worth of data. That amount of data would be suitable to use ARIMA to potentially forecast one year into the future, but with a six month prediction, the forecast will be more accurate. Furthering, the forecast could reasonably be recalculated every six months.

*Model evaluation procedure and error metric:*

For evaluation purposes, the AIC was compared between all tested ARIMA models. The combination of values that produced the lowest AIC was selected. That model, (1,1,0)(0,0,0)[0], also had a p-value less than 0.05.

Mean absolute error (MAE) was calculated for the model. MAE is an error metric describing an overall average of “the quantifiable difference between a measured value and its actual value.” (MathBlog, n.d.). A lower value will produce more accurate predictions. The MAE of the model was shown to be 0.357 which is indicative of a high rate of accuracy.

### E2: Annotated Visualization

*Screenshot of Annotated Visualization:*

A graph of blue and orange lines

Description automatically generated

### E3: Recommendation

My recommendation based on this analysis is to calculate the forecast every six months for the following six months. Furthermore, continue gathering daily revenue data to be able to enhance the accuracy of the forecasts over time. Six months was selected due to an increase in accuracy along with minimal devoted time per year to the analysis and potentially allow for planning and budget estimation.

## Part VI: Reporting

### F: Reporting

Jupyter Notebook was the IDE used in this project. The notebook was exported as an .html file.

*See Attached File:* d213task1complete.html

### G: Sources for Third-Party Code

Course materials. (n.d.)

Zach. How to Add a Trendline in Matplotlib (With Example). Statology. Published March 31, 2022. https://www.statology.org/matplotlib-trendline/

### H: Sources

Ahmed, I. (2023, May 31). *What are ACF and PACF Plots in Time Series Analysis?* Medium. https://ilyasbinsalih.medium.com/what-are-acf-and-pacf-plots-in-time-series-analysis-cb586b119c5d

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

GeeksforGeeks. (2020, November 24). AutoCorrelation. (2020, November 24). GeeksforGeeks. https://www.geeksforgeeks.org/autocorrelation/

Mathblog (n.d.). Absolute Error and Mean Absolute Error: Definition, Formula, Examples. (n.d.). MathBlog. https://mathblog.com/statistics/definitions/absolute-error/

Overload, D. (2023, June 17). Understanding ARIMA Models: A Comprehensive Guide to Time Series Forecasting. Medium. https://medium.com/@data-overload/understanding-arima-models-a-comprehensive-guide-to-time-series-forecasting-dfc7207f2406

Stephanie. (2016, February 25). Stationarity & Differencing: Definition, Examples, Types. Statistics How To. https://www.statisticshowto.com/stationarity/