Time Series Modeling – D213

Gooden, Nina S. Gooden [ID #: 009823504]

Dr. William Sewell

Time Series Modeling – D213

This analysis will explore the medical data of a theoretical real-world organization. By utilizing the time series modeling I’ve learned throughout the duration of this course, I will build revenue forecast to support the medical center’s concern about readmission in a data-first evaluation. I will also provide visualizations to support my assessments, in tandem with code for my models. As previously requested, I will also discuss the limitations and potential course of action this data supports.

## **Part I: The Research Question**

1. Research Question
   1. Using the ARIMA time series analysis, can we predict the medical center’s earning for the next 30 days?
2. Data Analysis Goals and Objective
   1. The primary goal of this analysis is to identify patterns in earned revenue to support the medical center’s evaluation of their readmission impact. I will be testing several versions of the available arima time series model creations in order to find one that is best suited for this analysis.

## **Part II: Justification for Technique**

1. Assumptions of time series model:
   1. Stationarity: Assumes that the mean, variance, and autocorrelation foundation of the data does not change over time. Data should be without trend, have a stable variance over time, and no seasonality. (Palachy, 2021)
   2. Autocorrelated data: Assumes that there is a degree of consistency or similarity between a selected time series and a legged-delayed version of that same time series. This is intended to compare the present value of success with that of the past. (Dotis-Georgiou, 2021)

## **Part III: Data Preperation**

1. Realization line graph
   1. In order to create the initial visualization of the time series, I had to import the data and run initial evaluations and preparations. As per the usual, I checked the head() function and searched for any null values as well.

A screenshot of a computer

Description automatically generated with medium confidence

* 1. I also had to convert the days time list to proper dates for the time series analysis. This was accomplished by defining a function and utilizing the datetime package. I set my target year to 2015, per the Medical Data Considerations and Dictionary scenario explanation. Code adaptation credit: (Daweo, 2021)

A screenshot of a computer

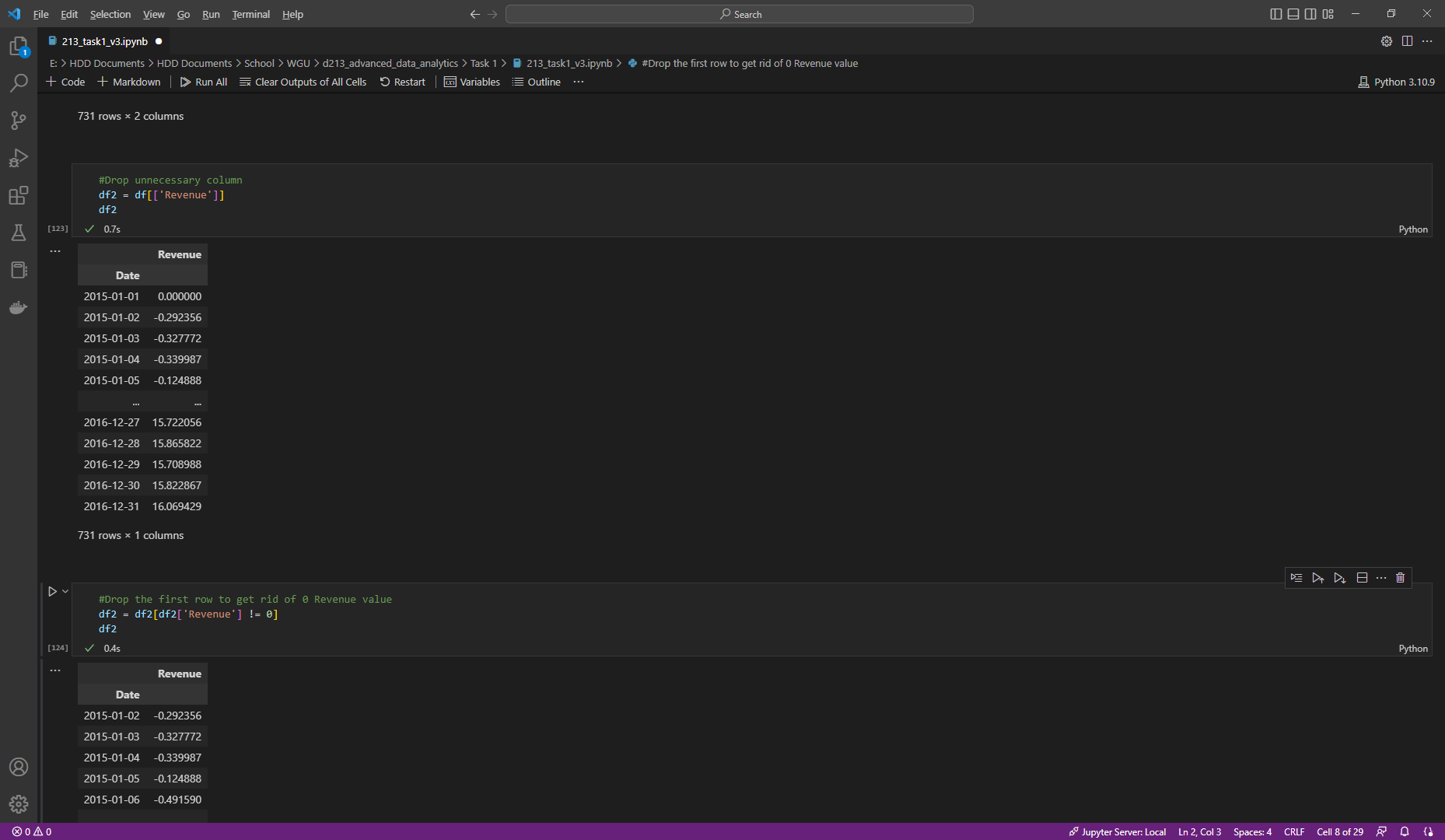
Description automatically generated with medium confidence

* 1. Due to the fact that I did not want the Date values changed in later evaluations (specifically the .diff() efforts), I set my index to take the Date values.

A screenshot of a computer

Description automatically generated with medium confidence

* 1. Lastly, I dropped the ‘Day’ column, as it was no longer necessary.



* 1. I also dropped the Day 1 data, as it resulted in a 0 value for Revenue and was not necessary for my analysis.

A screenshot of a computer

Description automatically generated

* 1. I created the line graph visualization.

Graphical user interface

Description automatically generated

1. Time step formatting
   1. Gaps in measurement: The data seems to follow a random walk with an upward drift. I also noted that there was no missing information or gaps.
   2. Length of sequence: After dropping the unnecessary Revenue value, the length of the sequence begins on Day 2, January 2nd 2015. It continues until December 31st 2016 for a sequence length of 730 days or two years.
2. Stationarity
   1. Training: In order to assess the dataset for stationarity, I first had to use the ADF. In doing so, I notes that the p-Value was greater than the critical values, so the data was non-stationary **(Kumar, 2021)**. I also noted the increasing trend.

A screenshot of a computer

Description automatically generated with medium confidence

* 1. Sanity checks to ensure the health of my data.

A screenshot of a computer

Description automatically generated

* 1. Next, I used the .diff() function to transform the dataset to stationary via differencing. The .diff() function subtracts each point from the previous value. This step also removed NA values after the differences were produced. It is worth noting that when I originally ran this evaluation, I had not indexed my Date column and this step through off every following interpretation.

Graphical user interface, text, application

Description automatically generated

* 1. The resulting data was visualized and found to be visually stationary.

Graphical user interface

Description automatically generated

* 1. I reran the ADF to reassess whether or not the dataset was stationary. The p\_Value is now less than .05 and the statistic value is less than the CV, so I was satisfied that the data was ready for the next step.

Graphical user interface, text

Description automatically generated

* 1. Test set split: I created the training and test kit while keeping the recommended 80/20 ratio. I also declared my random\_state and set shuffle to false so I could replicate my analysis.

A screenshot of a computer

Description automatically generated with medium confidence

* 1. I created the ACF visualization to look for any change in the data. I noted that the stationary series decomposes gradually, leading to a gentle decline.

Chart

Description automatically generated

* 1. I also created the PACF plot and noted a much more drastic drop. The trend is no longer present when compared to the original series, indicating the data has been properly prepared.

Graphical user interface, text, application

Description automatically generated

1. Preparing the data
   1. Detailed steps, including screenshot of code for training and test set split included in #3. Steps were as follows:
      1. Import necessary packages and libraries, as well as dataset.
      2. Determine health of overall data.
      3. Convert time series data to usable structure.
      4. Drop unnecessary data.
      5. Initial visualization analysis.
      6. Determine stationary vs. non-stationary status.
      7. Deploy differencing to transform data.
      8. Run ACF and PACF checks to ensure data is prepared for modeling.
2. Copy of cleaned Dataset
   1. Two datasets exported from this analysis.

Graphical user interface, text, application

Description automatically generated

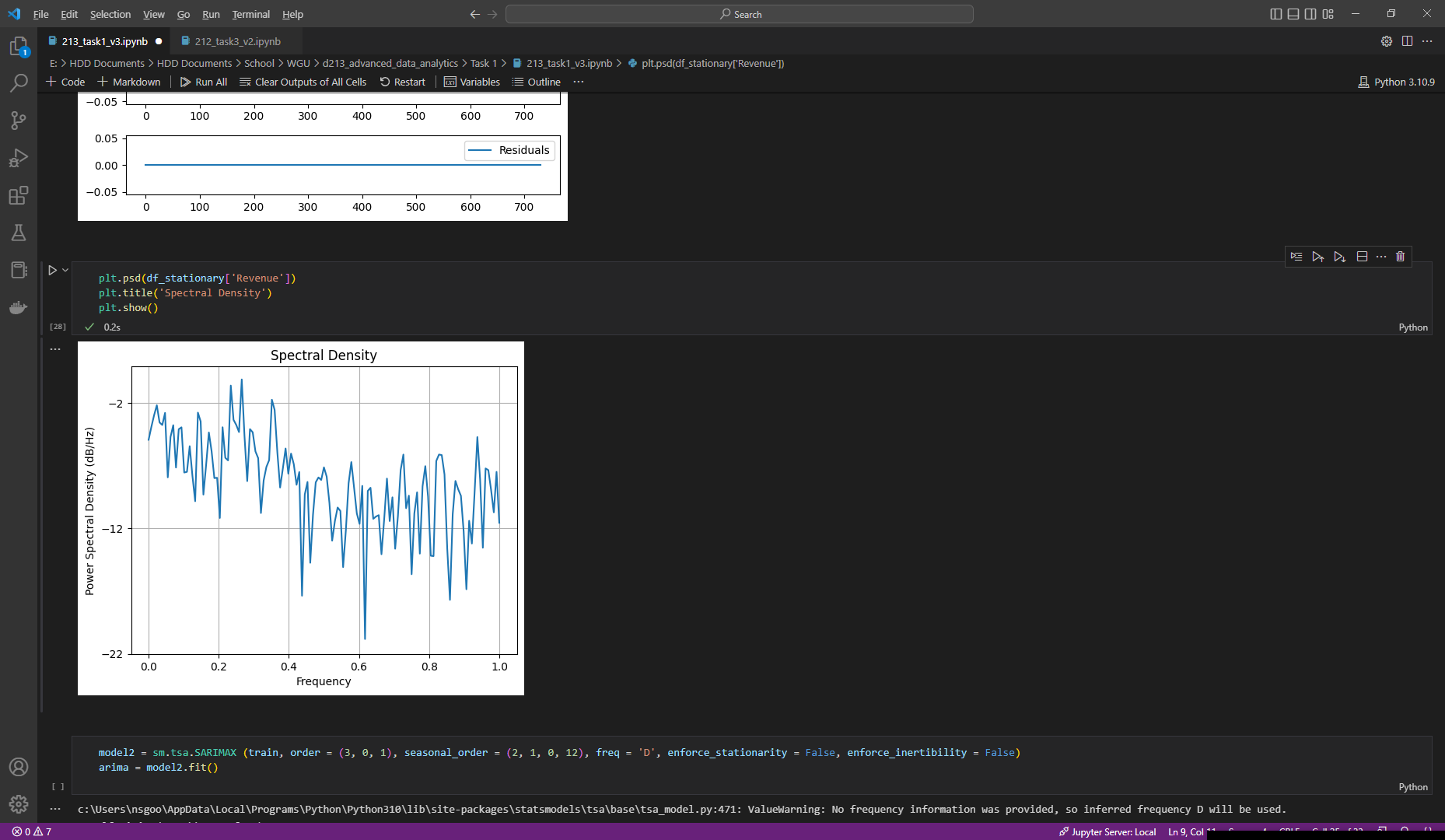
## **Part IV: Model Identification and Analysis**

1. Summarized findings and assumptions
   1. In order to summarize my findings and assumptions, I first created decomposing visualizations, which provided abstract models to confirm my previous observations.
   2. There is a slightly upward trend still.
   3. Here, I noted that seasonality was, indeed, a non-factor in the data.
   4. My residuals are a flat line, showing that there are none present.
   5. Autocorrelation was visualized in the previous ACF and PACF visualizations. The initial data was not stationary, however the differenced data is.

Graphical user interface

Description automatically generated

* 1. I also created a spectral density analysis to measure the power of the given signals, compared to frequency. I noted any weak or strong values, which included strong at 0.2-0.3Hz and weak just after 0.6Hz. This graph further exhibits no evidence of seasonal trends.



* 1. Selecting an ARIMA model required multiple tests of trial and error. I ended up needing to import the pmdarima library in order to use the auto\_arima function. The results returned a Best model value of ARIMA(1,1,0)(0,0,0)[0]. The second value indicates there is no seasonality in the data.

Graphical user interface

Description automatically generated

* 1. I used the value found with auto\_arima to create my ARIMA model and print the summary.

A screenshot of a computer

Description automatically generated with medium confidence

* 1. Since my research question is to make a prediction 30 days out, I targeted that number for my forecast model.

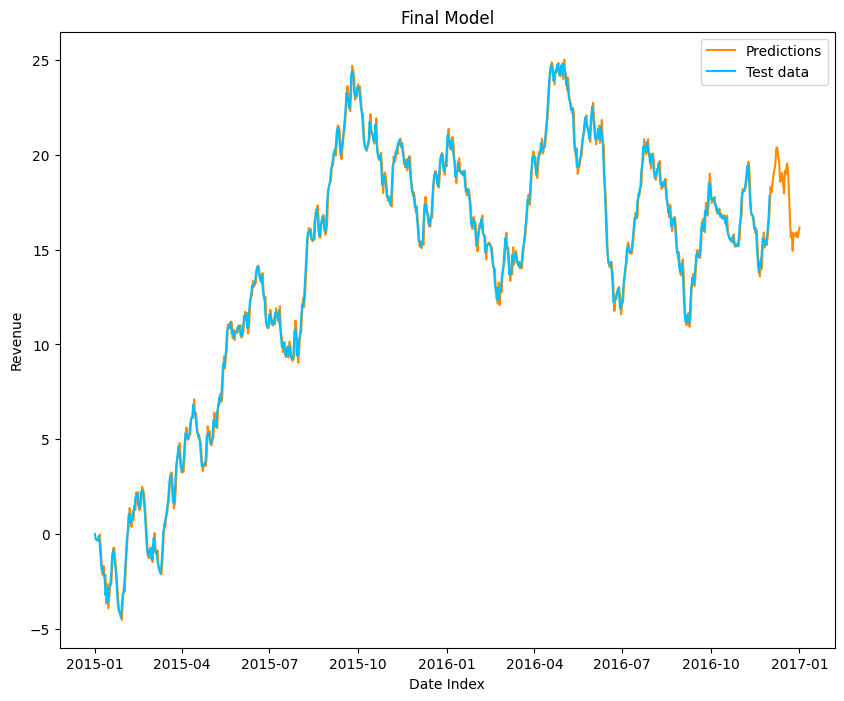
A screenshot of a computer

Description automatically generated

1. Visualization of the forecast
   1. I overlayed my forecast data over the trained dataset. The prediction extends 30 days past the given information, per my research question. Unfortunately, the accuracy of the model fails after 20 days, so the model will need to be rerun to bi-weekly to measure continued usefulness.

A screenshot of a computer

Description automatically generated with medium confidence



* 1. **I created a second graph of my updated predictions, which include the extended forecast information. I also created a visualization to show how the 30 day forecast compared to the held over test data (30 days). First, I recalled sliced my data.**

A screenshot of a computer

Description automatically generated

* 1. **Then I set the start and end variables based on those slices. I also created the pred model using the .predict() function.**

Text

Description automatically generated

* 1. **Which produced this (updated for clarity) visual.**

Chart, line chart

Description automatically generated

* 1. **Finally, I checked the accuracy of the model by finding the root mean square deviation. The result was 4.4%, sufficiently sized.**

Graphical user interface, text, application

Description automatically generated

* 1. **I fitted my last model.**

Graphical user interface, text

Description automatically generated

* 1. **And created my prediction fitted only to the training data.**

Text

Description automatically generated

* 1. **Before creating a visualization to reflect my predictions.**

Graphical user interface

Description automatically generated

1. Recommended course of action:
   1. Though the overall trend in the data is steadily increasing, there is a considerable downturn approaching, according to the model **(Pierre, 2021).** It is my recommendation that the medical center access spending during this time to lessen the impact of the revenue loss.
   2. In order to better understand whether or not this analysis is cause for action, the medical center will need to continuously run this analysis. It is my recommendation that a schedule be created to expand and review this information on a rolling basis.

## **Part VI: Reporting**

1. Development environment: Juypter Notebook

### G. Reference web sources

Sewell, W. (2022, July 19). *ARIMA PRESENTATION # 2*. Western Governors University. https://westerngovernorsuniversity-my.sharepoint.com/personal/william\_sewell\_wgu\_edu/\_layouts/15/

*Convert day numbers into dates in python.* (2021, August 3). Stack Overflow. https://stackoverflow.com/questions/68638178/convert-day-numbers-into-dates-in-python

*Converting year and day of year into datetime index in pandas.* (2015, December 14). Stack Overflow. https://stackoverflow.com/questions/34258892/converting-year-and-day-of-year-into-datetime-index-in-pandas

*TypeError: descriptor “date” for “datetime.datetime” objects doesn’t apply to a “int” object*. (2021, March 2). Stack Overflow. https://stackoverflow.com/questions/66431493/typeerror-descriptor-date-for-datetime-datetime-objects-doesnt-apply-to-a

pandas.DataFrame.set\_index — pandas 1.5.2 documentation. (n.d.). https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.set\_index.html

*TypeError: descriptor “date” for “datetime.datetime” objects doesn’t apply to a “int” object.* (2021b, March 2). Stack Overflow. https://stackoverflow.com/questions/66431493/typeerror-descriptor-date-for-datetime-datetime-objects-doesnt-apply-to-a

Time series / date functionality — pandas 1.5.2 documentation. (n.d.). https://pandas.pydata.org/docs/user\_guide/timeseries.html

GeeksforGeeks. (2020, November 16). Python - Convert day number to date in particular year. https://www.geeksforgeeks.org/python-convert-day-number-to-date-in-particular-year/

Real Python. (2021, August 20). Split Your Dataset With scikit-learn’s train\_test\_split(). https://realpython.com/train-test-split-python-data/

statsmodels.tsa.arima\_model.ARIMA.predict — statsmodels. (n.d.). https://www.statsmodels.org/v0.12.0/generated/statsmodels.tsa.arima\_model.ARIMA.predict.html

6.4.4.2. *Stationarity*. (n.d.). https://www.itl.nist.gov/div898/handbook/pmc/section4/pmc442.html

Descriptor “date” for “datetime.datetime” objects doesn’t apply to a “int” object | bobbyhadz. (n.d.). Blog - Bobby Hadz. https://bobbyhadz.com/blog/python-typeerror-descriptor-date-for-datetime-datetime-objects

Kumar, J. (2021, December 16). How to Check if Time Series Data is Stationary with Python? from How to Check if Time Series Data is Stationary with Python? https://www.geeksforgeeks.org/how-to-check-if-time-series-data-is-stationary-with-python/

#### H. Acknowledge sources

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

Palachy, S. (2021, December 9). Stationarity in time series analysis - Towards Data Science. Medium. https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322

Dotis-Georgiou, A. InfluxData. (2021, December 10). InfluxDB: Open Source Time Series Database. https://www.influxdata.com/blog/autocorrelation-in-time-series-data/