HIGH SCHOOL EDITION
Journal of Student Research
Volume 11 Issue 3 (2022)

Predicting Flood Streamflow with Auto Regressive Integrated Moving Average Models

Everest Yang[1] and Yong Zhuang[#]

[1] Lexington High School, Lexington, TX. USA
[#]Advisor

ABSTRACT

Flooding is the most common natural disaster and continues to increase in frequency and intensity due to climate changes [7]. Currently, there is a lack of efficient tools to predict flooding. This research is aimed to create a Time Series Machine Learning (ML) program using Auto Regressive Moving Average (ARIMA) models to forecast streamflow, one of the most prominent factors in flood prediction. A streamflow dataset from the Ganges River, Bangladesh was used to plot several graphs of the river Log Volume to observe possible trends. Another plot was graphed to check and quantify how much the distribution of the stream volume changed over the course of 10 years using KL Divergence. The plot analyses and Partial Autocorrelation Function (PACF) and Autocorrelation Function (ACF) tests were used to help obtain the ARIMA parameters of (p, d, q) as (1, 1, 1). However, the forecasted streamflow of the ARIMA function was not accurate when compared with previously recorded data because of heavy seasonality. As a result, the final program was redesigned with Seasonal ARIMA (SARIMA) to account for the inaccuracy. The SARIMA model was used to forecast the streamflow of subsequent years and was close to the actual recorded data. Such accuracy indicates that this method can be a useful tool in navigating and preparing for floods.

Introduction

A flood is an overflow of water that submerges dry land and is primarily caused by heavy rainfall, overflowing rivers, melting snow (ice), and collapsed dams. According to the World Health Organization, floods affected more than 2 billion people worldwide between 1998 and 2017 [7]. Bangladesh is among the worst affected, with high human and material losses. Currently, hydrologists attempt to predict floods by tracking storm conditions and rainfall data. They also use wireless electronic sensors placed in and near bodies of water to measure water levels and drainage in real time. Unfortunately, sensor data isn't available in many locations, so scientists are constantly exploring ways to extend how far in advance they can forecast floods [1]. Because flooding is such a common and destructive natural disaster, this research aims to counter the problem with ARIMA models, an effective statistical analysis model in ML that uses time-series data to predict future trends.

Machine Learning is a subset of Artificial Intelligence focused on building systems. It can learn from historical data, identifying patterns to make logical decisions with little to no human intervention. ML is growing in importance due to increasingly enormous volumes of data and the greater access to computational power provided by high-speed internet [16]. In this study, a dataset containing streamflow at the Hadin Bridge on the Ganges River in Bangladesh was used as historical data for the ARIMA model to interpret. The U.S. Geological Survey uses the term streamflow to refer to the amount of water flowing in a river [15]. Before forecasting, various plot analyses and visualization sequences were conducted to observe for possible trends. Next, the ARIMA parameters (p, d, q) were obtained with several ML techniques: PACF plots to find “p”, ACF plots to find “q”, and differencing between the two functions for “d” [9, 11]. The three parameters were then imputed into the ARIMA algorithm for streamflow prediction. It is important to note that the finalized program was

redesigned multiple times to account for heavy seasonality in the dataset, which led to autocorrelation and influenced the accuracy.

Methodology

Part 1: ARIMA Overview, Materials and Procedure

The ARIMA model can be split into three parts, "AR-I-MA", which correspond to the three parameters of (p, d, q). The "AR" stands for Auto-Regressive, and it is a model dependent on its own lagged observations. The "p" parameter is the order of the "AR" term, also called the lag order. The "MA" stands for Moving Average and is a model that depends on its lagged forecast errors. The "q" parameter is the order of the MA term, also called the order of moving average. The "I" stands for the Integration between the AR and MA models. It uses the differencing of raw observations to make the time series data stationary. The "d" parameter is the order of the "I" term, also called the degree of differencing. The objective is to find the (p,d,q) parameters to input into the ARIMA algorithm for predictions [4, 11].

Among the required materials for this project is a computer with an operating system that can download the Jupyter Notebook IDE and the following library packages: Matplotlib, Pandas, Numpy, seaborn, scikit, SciPy, statsmodels, pmdarima. A composition lab notebook (to record the process) and a streamflow dataset are also required. In this study, I coded with Windows and used cmd to run Jupyter Notebook on Google Chrome. I also used an excel file provided by my mentor, Yong Zhuang, that contained flow data observed from the Hadin Bridge on the Ganges River, Bangladesh.

The dataset includes recorded river streamflow from almost every day between 1/1/1967 to 12/31/2017 with over 18,600 data points (although there are some missing values). The streamflow was measured in "Q", a metric for stream-discharge volume in cubic meters per second. First, save the excel file as a Comma-separated values (CSV) file and clean the data to ensure it is in the proper syntax (I recommend modifying the dataset into only two columns: time as the index (x-axis), and Q (m^3) as the streamflow (y-axis)). A CSV is a delimited data format that has columns separated by commas and allows the IDE to read the file with greater ease [5]. Next, open Jupyter Notebook and import the necessary libraries in the first cell block; use Pandas to read the CSV file. I specifically chose to use Jupyter Notebook because it is an open-source web IDE that allows data scientists to create documents with live code, equations, computational output, and visualizations. It is also a popular environment for machine learning [17].

Part 2: Data Visualization and Analysis

Conducting various plot analyses and visualization sequences before the forecasting process is a crucial step in data science. This helps observe possible trends and allows for a better understanding of the graph shape. Thus, I plotted several graphs of the original dataset to serve as a guide in building the actual model. Using the required python libraries to plot the original dataset (Figure 1), it is observed that there are some missing values in the dataset (Figure 2). Therefore, this research used the ten most recent years with no missing values (2007-2016) as plot data. All the graphs in this section were plotted with Matplotlib and visually improved with seaborn.

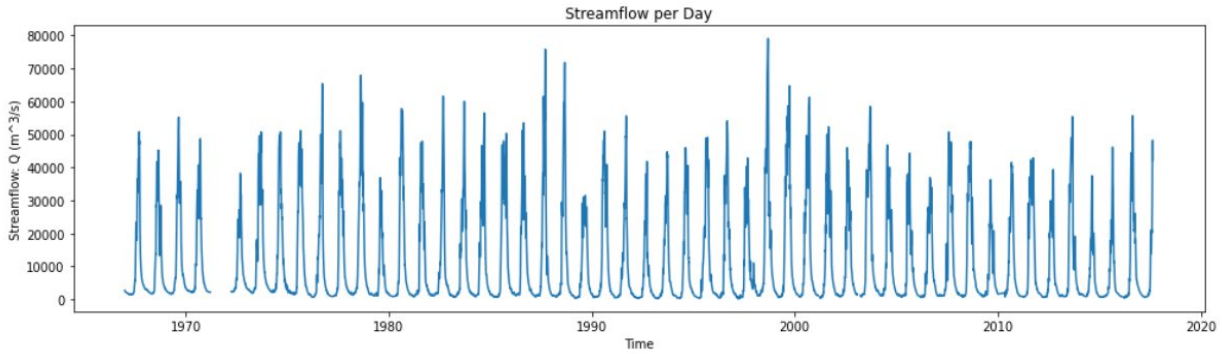

Figure 1. Graph of the original dataset – Streamflow per Day.

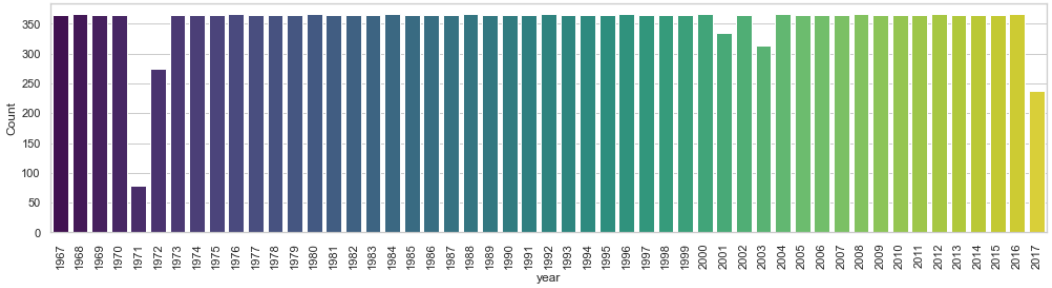

Figure 2. Graph depicting which years contain missing values (unrecorded data). As shown, 1971, 1972, 2001, 2003, and 2017 contain missing values.

I then plotted the Log Volume of the most recent ten years with no missing values shown by Figure 3 below. There are two main reasons to use logarithmic scales in charts and graphs. The first is to respond to skewness towards large values, i.e., cases in which one or a few points are much larger than the bulk of the data. The second is to show percent change or multiplicative factors. This gets rid of any obscure outliers that can alter the program's accuracy.

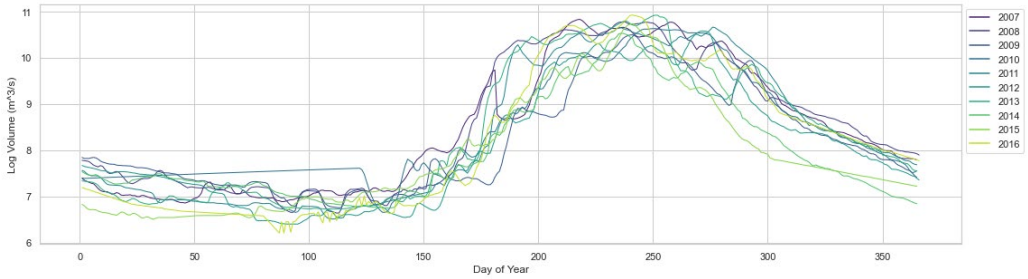

Figure 3. Log Volume of the most recent ten years with no missing values.

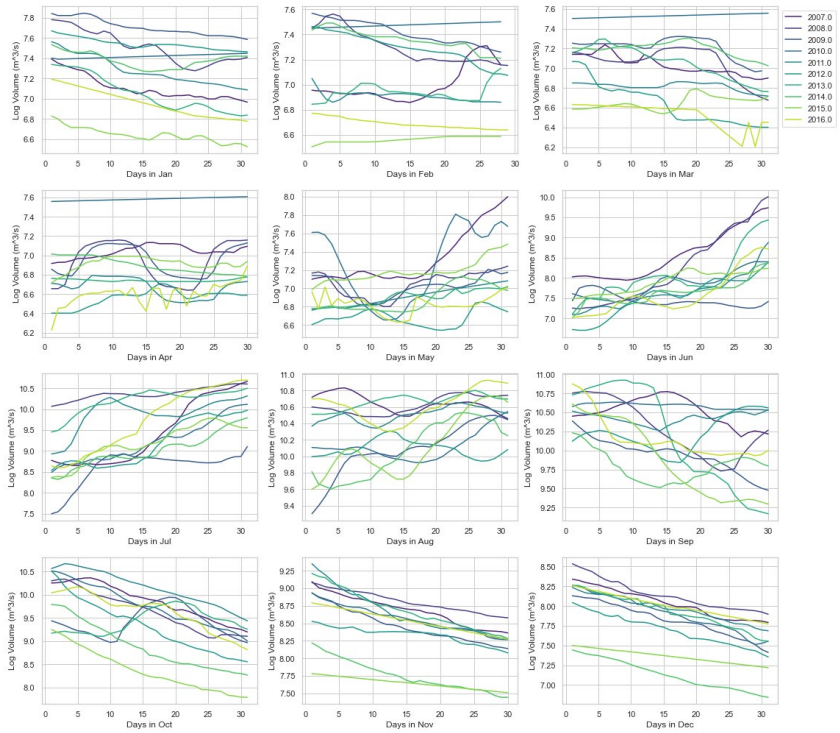

Figure 4. Log Volume of the most recent ten years with no missing values filtered by months.

Figure 4 displays the same data as Figure 3 but is filtered by month for a more specific analysis that reveals unique patterns. For example, the Log Volume of May and September are more randomized with no visible patterns, while data from November and December display more linear behaviors.

Next, I plotted the Log Volume vs Density shown by Figure 5 and Figure 6 below. Kernel Density Estimation is a way of estimating the probability distribution function in a non-parametric manner (non-parametric method refers to a set of techniques where no assumptions about the distribution are made and estimation is done without calculating the sample statistics like mean, standard deviation, etc.) [14]. The graph of Density vs Log Volume compares the distribution of river flow over multiple years. It must be noted that the density of a distribution refers to how much probability is distributed over data, and not the actual probability. To calculate the probability for a continuous random variable, one must find the area under the curve between the points of consideration <a,b>. At times, the graph can come across as positively skewed and long, heavy-tailed distributions (dense in the beginning but with probability distributed over very large values). In such scenarios it is a common practice to take the Logarithm of the data under consideration and such distributions are called Log-Normal.

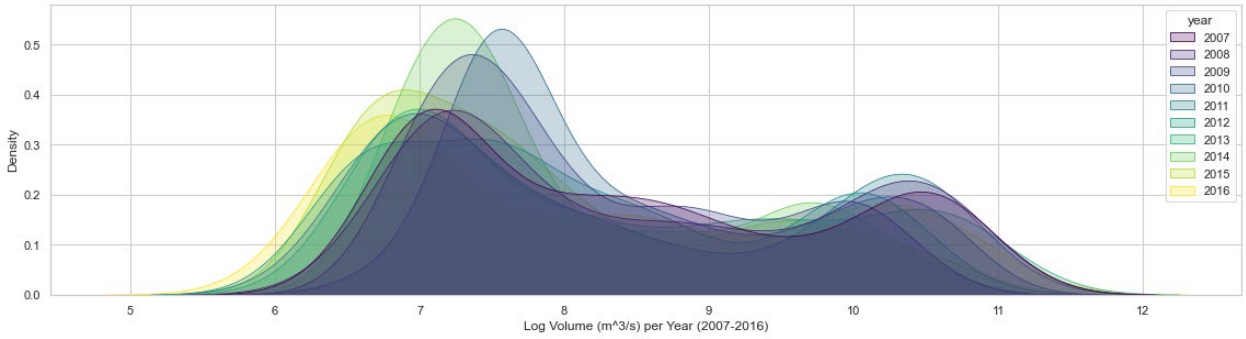

Figure 5. Log Volume vs Density per year.

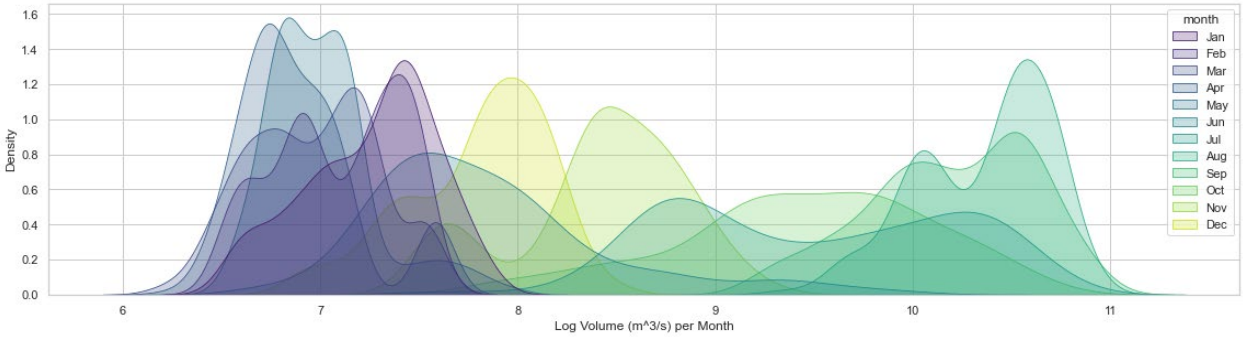

Figure 6. Log Volume vs Density per month.

The last graph (Figure 7) in this section accounts for changes in the distribution of stream volume over the course of 10 years. I used KL Divergence, a distance metric (it is not a distance/true metric per say because of non-symmetric behavior) that helps in measuring the distance between two probability distributions [3]. Intuitively, this measures how much a given arbitrary distribution differs from the true distribution.

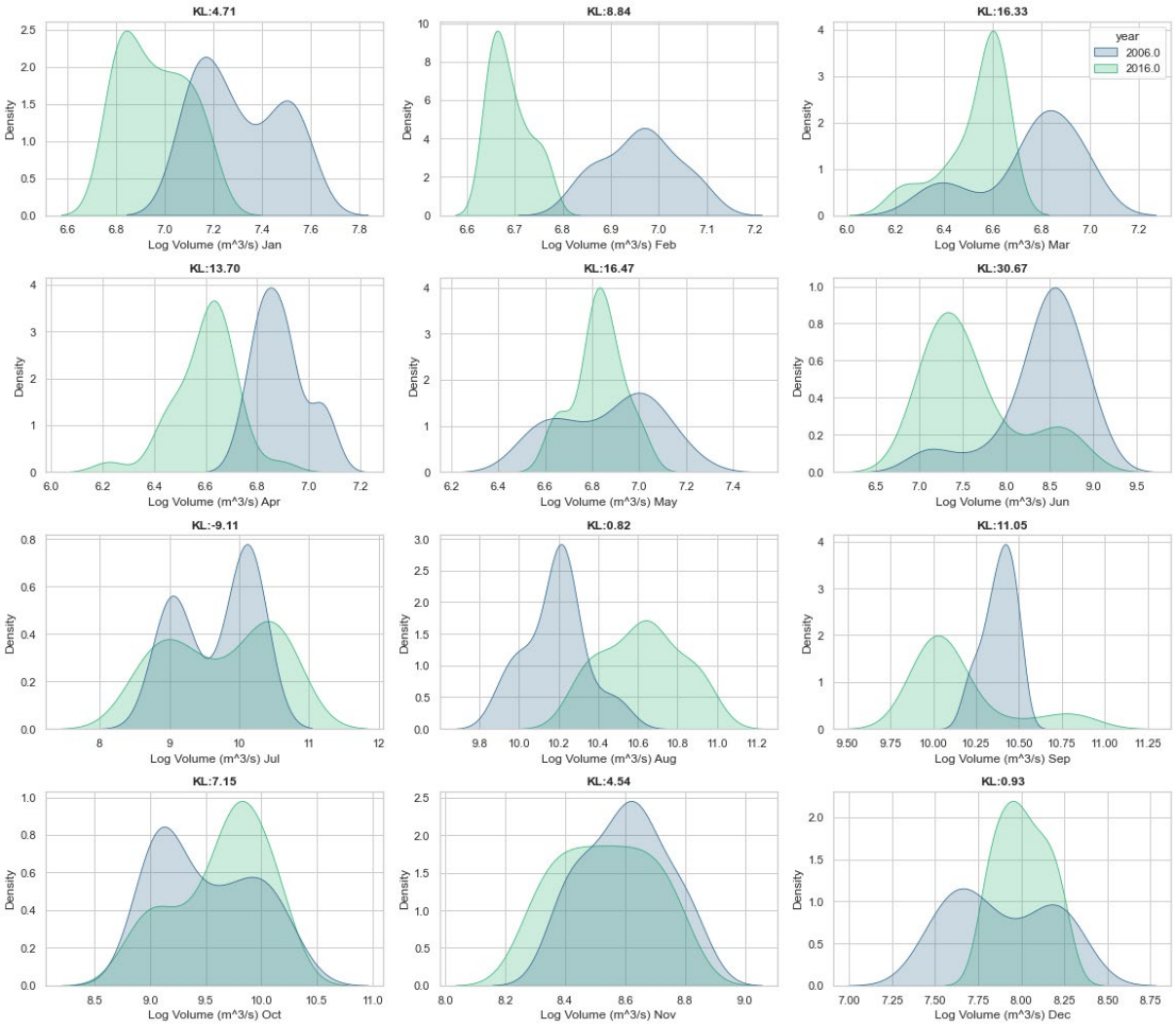

Figure 7. Comparison of stream volume in a 10-year span between 2006 and 2016.

Figure 7 shows that the shaded blue graph is streamflow data from 2006, while the shaded green graph is streamflow data from 2016. KL Divergence is effective to see how the log volume has changed in a ten-year span. For example, the average January 2006 Log Volume is greater than in January 2016. On the other hand, the two plots swap places in August which emphasizes how much the streamflow has changed over these ten years.

Part 3: Parameters Determination and Experiments

With a much better understanding of the streamflow data after conducting "Part 2: Data Visualization and Analysis", I began constructing the ARIMA model. Since there are over 18,600 values in the dataset, I took the monthly (rather than the daily) average streamflow to avoid unnecessary noise deterring the program shown by Figure 8. Nevertheless, the resulting shape of the monthly average was still very similar to that of the daily streamflow.

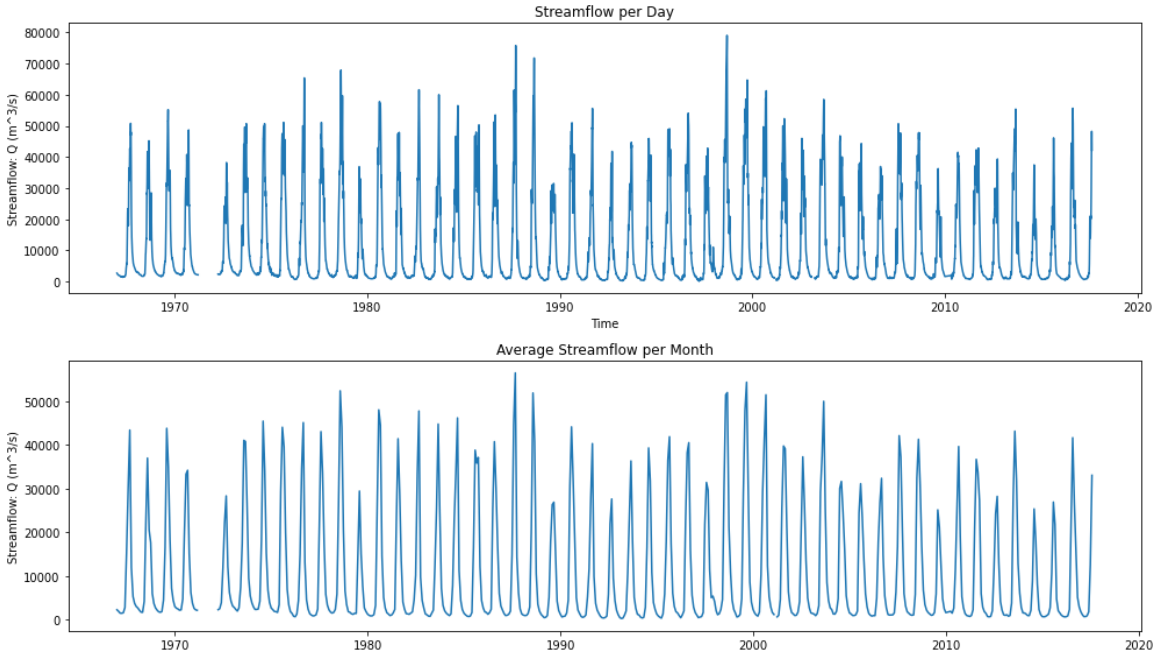

Figure 8. Comparison between the original dataset of Streamflow per Day vs the Average Streamflow per Month.

It is crucial to discern whether the data is stationary or non-stationary to build the ARIMA model. For stationary time series, properties do not depend on time, which makes it much easier to forecast future values. On the other hand, time series with trends or seasonality are not stationary and can impact the value of the series in various places [6]. The first test I conducted was the Augmented Dickey Fuller Test (ADF), a common statistical test used to test whether a given Time series is trend stationary or not [11, 12].

The ADF test uses the adfuller() function in statsmodels.tsa.stattools from the statsmodel library. It returns the p-value, the value of the test statistic, number of lags considered for the test, and the critical value cutoffs. When the test statistic is lower than the critical value and the p value is less than 0.05, the null hypothesis is rejected and it can be inferred that the time series is stationary [11, 12]. In this study, the test statistic was -5.0002, the p-value was 2.2e^-5 (approximately 0.0148), the number of lags considered was 13, and the critical values were {'1%': -3.4416553818946145, '5%': -2.8665274458710064, '10%': -2.5694261699959413}. Thus, the data is trend stationary.

The next step is to find the 3 parameters of the ARIMA function. Since ARIMA is the AR model integrated with the MA model, I found the AR parameter, "p", with the PACF plot and the MA parameter, "q", with the ACF plot. Mohammad Masum states that ACF is the complete autocorrelation function and explains how the present value of a given time series is correlated with the past. PACF is the partial autocorrelation function that explains the partial correlation between the series and lags itself. Simply, PACF uses linear regression to predict y(t) from y(t-1), y(t-2), and y(t-3) [11].

After importing the "plot_acf" and "plot_pacf" functions from the "statsmodels.graphics.tsaplots" library, I plotted Figure 9 which portrays the ACF and PACF plot of no differencing with lag= 24. As shown, both plots are not stationary because the data points are not close to the x-axis; rather, the ACF plot has a sinusoidal shape and the PACF plot is random. Looking closely at the ACF plot, the first value and the 12th value have a high autocorrelation. Likewise, the 6th and 18th observations are highly correlated. This means that a very similar value is found every 12 months, or 1 year [10].

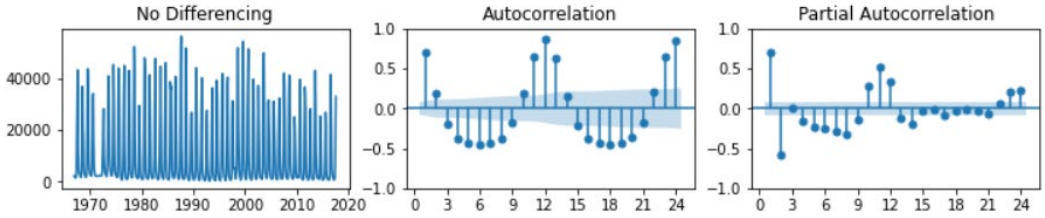

Figure 9. ACF and PACF plots of no differencing with lag= 24.

Thus, I conducted more ACF and PACF plots, experimenting with the differencing and the lag values. Figure 10 below shows the 1st and 2nd differencing of multiple lag values, all which are factors of 12. The plots with the points closest to the x-axis (the most stationary) are the 1st differencing with lag=36 and 2nd differencing with lag=72. This research used the 1st differencing with lag=36 to build the ARIMA model because it would be just as accurate as the 2nd differencing but take much less time to train.

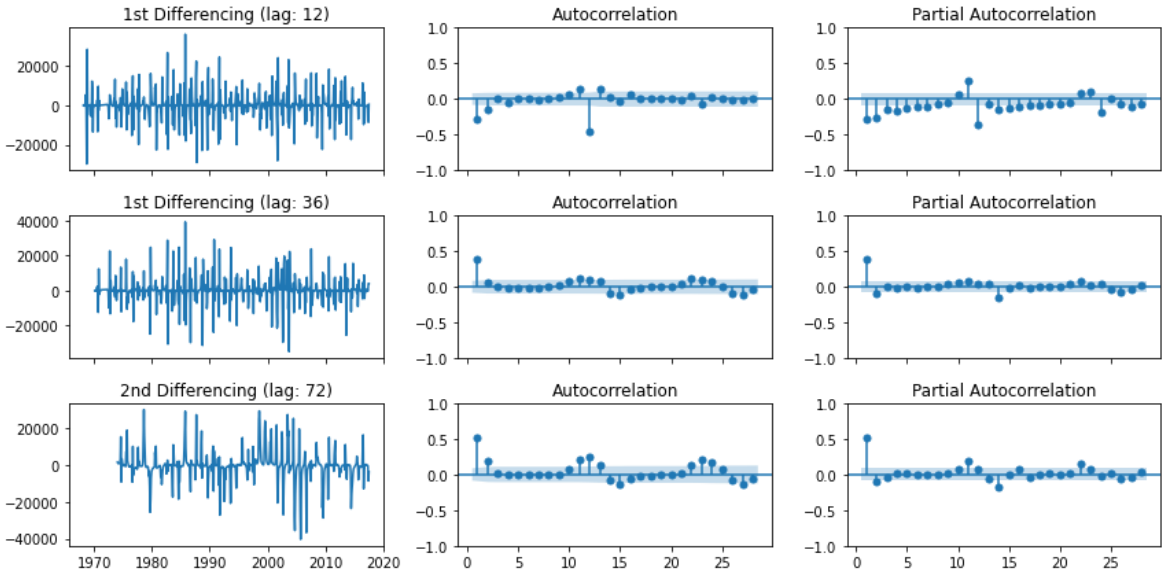

Figure 10. ACF and PACF plots of 1st and 2nd differencing with several lag values: 12, 36, 72.

Results

In the final analysis, the obtained ARIMA parameters of (p, d, q) are (1, 1, 1). This study used the 1st order difference (d=1) since it has the least number of significant lags in both the ACF and PACF plots. Then, PACF is significant at lag=1, therefore (p=1). ACF is also significant at lag=1, therefore (q=1). So, theoretically, (p=1, d=1, q=1) will train the ARIMA function. However, the function will not work as well as intended because it fails to address the issue of "autocorrelation", an important assumption in parametric statistical modeling [10]. As shown by Figure 11 below, the predicted forecast (orange curve) is not particularly accurate when compared to previously recorded data (blue curve). This is due to heavy seasonality in the dataset, the presence of variations that occur at specific regular intervals within a year [10]. The portrayal of sinusoidal waves in the output of the graph proves that the streamflow is heavily dependent on time. It tends to drastically increase during the summer months, which are characterized by heavy precipitation, and drastically decrease during colder seasons, which experience less precipitation. Therefore, I redesigned the program in Seasonal ARIMA (SARIMA).

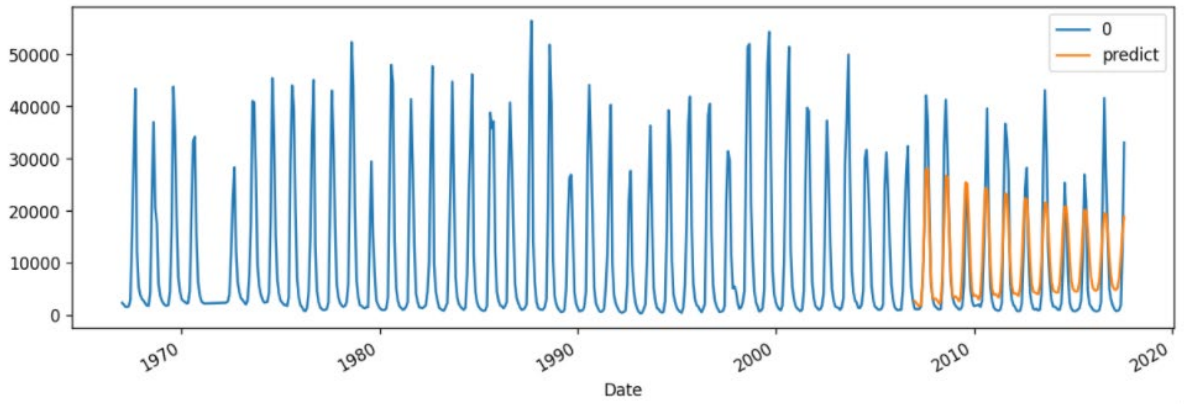

Figure 11. Forecasted streamflow with the default ARIMA algorithm.

SARIMA or Seasonal ARIMA, is a variant of ARIMA that explicitly supports univariate time series data with a seasonal component. It adds three new hyperparameters to specify the autoregression (AR), differencing (I), and moving average (MA) for the seasonal component of the series, as well as an additional parameter for the period of the seasonality [2, 18]. Knowing p=1, d=1, q=1 from earlier, I needed to find “m”, the period of seasonality in the data. As shown by the graph of the original dataset from part two of the methodology, the sinusoidal shape has a cyclic pattern. This can be explained by the fact that floods are more frequent during the spring, when there is more rainfall and snowmelt than in the fall. Since this cycle repeats every year, or every 12 months, the parameter of “m” equals 12.

This research used the SARIMA parameters (1, 1, 1, 12) to conduct two tests. The first test split the original dataset into two parts: one part to train the model and the other part to compare with the forecasted streamflow. In the first test, the data was split into 1967-2007 and 2008-2017. I used the historical data from 1967-2007 to train the SARIMA model, which was used to forecast the streamflow from 2008-2017. I then compared the predicted streamflow to the actual data from 2008-2017. Figure 12 below reveals the relatively high accuracy of this program, as the sinusoidal shapes match and are similar in values. The actual streamflow is higher at times (such as in 2013), but that could be the result of special outliers, making it more difficult to predict since this model only considers streamflow.

The first test:

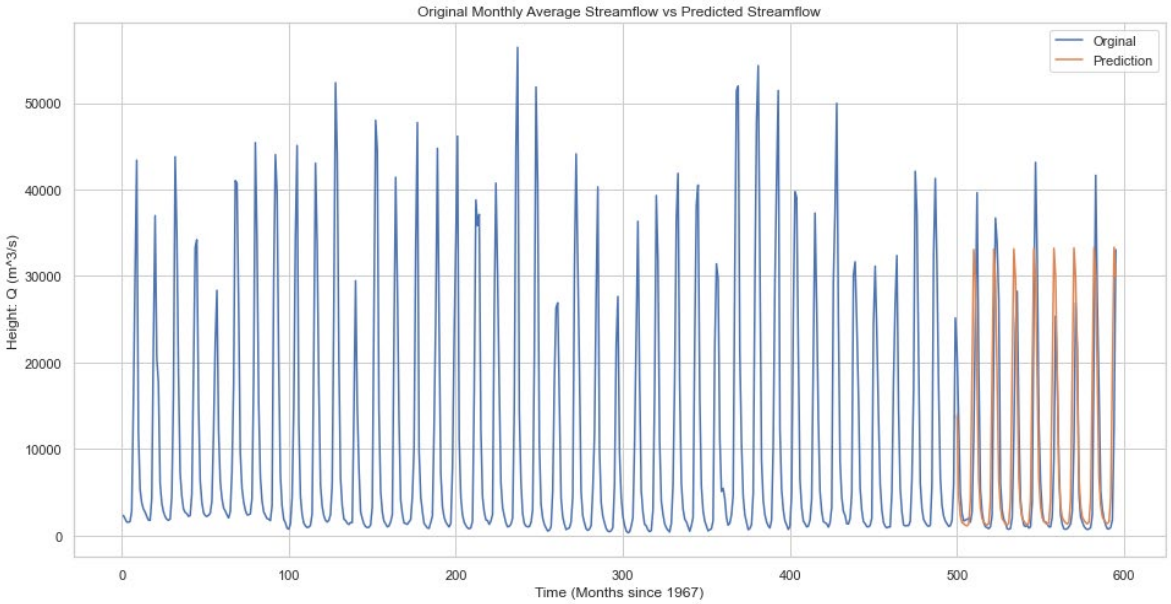

Figure 12. Trained SARIMA model with historical data from 1967-2007 to forecast the streamflow from 2008-2017. Then compared the predicted streamflow to the actual data from 2008-2017.

Now that we know the accuracy of the SARIMA model, I conducted the second test by using the entire original dataset of 1967 to 2017 to forecast the streamflow from 2018 to 2024 (the results of which are displayed below in orange). As shown, there is a slow upward trend with a similar cyclic pattern as the test before. This forecast can be used as a baseline of the streamflow of the Ganges River. Because the original dataset didn't have data post 2017, the accuracy of the forecast couldn't be determined yet. Once the actual data becomes available, it can be compared with the forecast to determine the accuracy. The foundation of this research rests on the ability to use ML to forecast future streamflow of a river and ultimately predict flooding.

The second test:

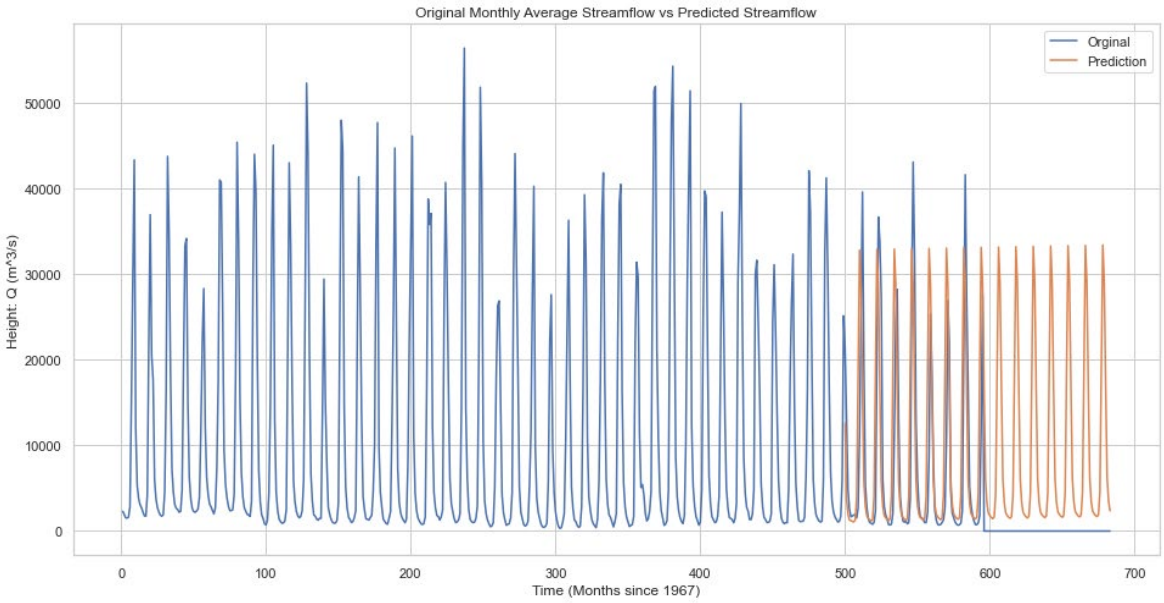

Figure 13. Trained SARIMA model with historical data from 1967-2017 to forecast the streamflow between 2018-2024.

Discussion & Conclusion

This study used a streamflow dataset of the Ganges River to forecast future river streamflow with ML algorithms. Originally, the default ARIMA forecasting was not precise enough to accurately detect flooding because the program had auto-correlation caused by heavy seasonality in the dataset. Although this problem was solved with SARIMA, there are other limitations preventing the program from reaching full accuracy, which are discussed in the next section. Nonetheless, this project provides a ML solution to help governments and local officials forecast river streamflow and use it as a baseline to predict and prepare for floods ahead of time. This could potentially save money, preserve resources, prevent casualties, and perhaps also benefit farmers who monitor flood behavior to tend their crops (i.e., redirecting irrigation and preparing for droughts). With ML, there is an abundance of potential for this program to become a useful tool for flood forecasting. Currently, algorithms are being improved to better process complex data sets and predict conclusions that could be widely beneficial. Future research will be required to address the limitations and improve this program.

Future Research/Limitations

Though this project has an abundance of potential, there are several limitations. Firstly, the dataset in this study only contains 18,600 values, which is an adequate amount. However, gathering more data, particularly from recent years, would increase the accuracy of the forecasted streamflow because larger sample sizes provide more precise estimates. Secondly, instead of coding in Python, switching to R could refine the data analysis since many Python libraries and functions contain consequential limitations. R is known to be widely applicable and highly useful for data science, specifically machine learning. Furthermore, streamflow is only one of many aspects that contribute to flooding. The dataset from Bangladesh involves a number of other factors: cyclones, abundance of low-lying land, melting water from the Himalayas, increasing urban areas, heavy deforestation and heavy monsoon rains [8, 13]. Pollution is also included amongst that list, as the Ganges River is considered as one of the most polluted waterways in the world. The climate in Bangladesh is unique from other locations, so streamflow could also be influenced by induced global warming, a surplus of precipitation, deviations from typical climate conditions, etc. To create a program that is nearly 100% accurate is arduous and would require masses of data, time, funds, and labor. Lastly, although this research accounted for some of the heavy seasonality with SARIMA, some trend stationarity hidden within the dataset will be difficult to extract. I will eventually use a statistical model to calculate the forecast accuracy that can be applicable to all streamflow datasets, not just the one used in this research. By addressing these limitations, I hope to establish improvements that enable this program to save millions of lives from disastrous floods.

Acknowledgments

Special thanks to my mentor/advisor Yong Zhuang, a Ph.D. in applied machine learning at UMass Boston. Yong specializes in Deep learning, Spatio-temporal analysis, time series forecasting, and feature selection. He provided me with the dataset containing the streamflow of the Ganges River, Bangladesh used in this research. The dataset is from a collaboration between Tufts University's Environmentalist team and the UMass Knowledge Discovery Lab (KDLab), which specializes in data analysis.

References

[1] Brewster, Signe. "Flickr tags could help predict floods." *Science.org*, 7 Mar. 2017, www.science.org/content/article/flickr-tags-could-help-predict-floods. Accessed 4 July 2022.

[2] Brownlee, Jason. "A Gentle Introduction to SARIMA for Time Series Forecasting in Python." *Machine Learning Mastery.com*, 17 Aug. 2018, machinelearningmastery.com/sarima-for-time-series-forecasting-in-python/#:~:text=to%20use%20SARIMA.-,What%20is%20SARIMA%3F,data%20with%20a%20seasonal%20component. Accessed 4 Feb. 2022.

[3] ---. "How to Calculate the KL Divergence for Machine Learning." *Machine Learning Mastery*, 19 Oct. 2018, machinelearningmastery.com/divergence-between-probability-distributions/. Accessed 4 Feb. 2022.

[4] ---. "How to Create an ARIMA Model for Time Series Forecasting in Python" ["How to Create an ARIMA Model for Time Series Forecasting in Python"]. *Machine Learning Mastery*, edited by Jason Brownlee, 9 Jan. 2017, machinelearningmastery.com/arima-for-time-series-forecasting-with-python/. Accessed 21 Oct. 2021.

[5] "CSV, Comma Separated Values (RFC 4180)." *loc.gov*, Library of Congress, 11 Feb. 2020, www.loc.gov/preservation/digital/formats/fdd/fdd000323.shtml. Accessed 4 July 2022.

[6] "8.1 Stationarity and Differencing." *Otexts*, otexts.com/fpp2/stationarity.html. Accessed 4 Feb. 2022.

[7] "Floods." *World Health Organization*, WHO, www.who.int/health-topics/floods#tab=tab_1. Accessed 21 Oct. 2021.

[8]Hasnet, Saif, and Mike Ives. "Bangladesh Floods Cause Death and Destruction in Sylhet." *The New York Times*, New York Times Company, 24 June 2022, www.nytimes.com/2022/06/24/world/asia/sylhet-bangladesh-floods.html. Accessed 4 July 2022.

[9] Masum, Mohammad. "Time Series Analysis: Identifying AR and MA using ACF and PACF Plots." *Towards Data Science*, 13 Aug. 2020, towardsdatascience.com/identifying-ar-and-ma-terms-using-acf-and-pacf-plots-in-time-series-forecasting-ccb9fd073db8. Accessed 4 Feb. 2022.

[10] Peixeiro, Marco. "The Complete Guide to Time Series Analysis and Forecasting" ["The Complete Guide to Time Series Analysis and Forecasting"]. *Towards Data Science*, edited by Maxo Peixiero, 7 Aug. 2019, towardsdatascience.com/the-complete-guide-to-time-series-analysis-and-forecasting-70d476bfe775?gi=b05b1ecae15a. Accessed 21 Oct. 2021.

[11] Prabhakaran, Selva. "ARIMA Model – Complete Guide to Time Series Forecasting in Python" ["ARIMA Model – Complete Guide to Time Series Forecasting in Python"]. *Machine Learning +*, edited by Selva Prabhakaran, 21 Aug. 2021, www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/. Accessed 21 Oct. 2021.

[12] ---. "Augmented Dickey Fuller Test (ADF Test) – Must Read Guide" ["Augmented Dickey Fuller Test (ADF Test) – Must Read Guide"]. *Machine Learning Plus*, 2 Nov. 2019, www.machinelearningplus.com/time-series/augmented-dickey-fuller-test/. Accessed 9 Dec. 2021.

[13] "Rivers and Flooding Case Study: Bangladesh." *BBC*, www.bbc.co.uk/bitesize/guides/zgycwmn/revision/4#:~:text=Causes%20of%20flooding%20in%20Bangladesh&text=Melt%20water%20from%20the%20Himalayas,Increasing%20urban%20areas. Accessed 4 July 2022.

[14] Saeed, Mehreen. "Kernel Density Estimation in Python Using Scikit-Learn." *StackAbuse.com*, stackabuse.com/kernel-density-estimation-in-python-using-scikit-learn/. Accessed 4 Feb. 2022.

[15] "Streamflow and the Water Cycle." *The U.S. Geological Survey (USGS)*, Water Science School, 12 June 2019, www.usgs.gov/special-topics/water-science-school/science/streamflow-and-water-cycle. Accessed 4 July 2022.

[16] "What is Machine Learning?" *Micro Forum*, www.microfocus.com/en-us/what-is/machine-learning. Accessed 4 July 2022.

[17] "Why You Should be Using Jupyter Notebooks." *odsc.medium*, ODSC - Open Data Science, 15 July 2020, odsc.medium.com/why-you-should-be-using-jupyter-notebooks-ea2e568c59f2#:~:text=The%20Jupyter%20Notebook%20is%20an,text%20in%20a%20single%20document. Accessed 4 July 2022.

[18] Yiu, Tony. "Understanding SARIMA (More Time Series Modeling)." *Towards Datascience.com*, 30 Apr. 2020, towardsdatascience.com/understanding-sarima-955fe217bc77. Accessed 4 Feb. 2022.