“WEBSITE TRAFFIC FORECASTING USING MACHINE LEARNING”

PROJECT REPORT

Submitted to University of Kerala in partial fulfilment of the requirements for the award of

MASTER OF SCIENCE (COMPUTER SCIENCE)

PROJECT CODE : CS 543

UNIVERSITY OF KERALA

THIRUVANANTHAPURAM

2021 - 2023

Contents

Chapter 1

Introduction

People who work for web service providers need to know how much traffic a web server is getting, because if they don't, customers might have to wait a long time and leave the site. However, this is a difficult task because it requires making accurate predictions about how people will act based on their randomness. In this project, I show how to build an architecture that takes source data and uses it to make predictions about how many people are going to see a given page at a given time. Depending on the website's response, web applications handle HTTP GET requests, media apps spread content based on what the user wants, and so on and so forth. Request time will have a big impact on how the end-user sees the quality of the service.

A lot of people have left a lot of platforms because they took too long to respond. However, the response time is the time between when the application receives the request and when it sends back the answer. This is called the response time. This can't be taken away. In the case of web services, the response time is too long for customers to expect. Developers have been able to figure out when the response time is too long, known as web congestion. A time series with the dates and number of page views make sense for the problem. The purpose of this research is to design a forecasting model to predict web traffic based on the certain features like page name, visited date and the number of visits for pages for a year. As more people gain access to the internet around the world, the increase in traffic to practically all websites have become unavoidable. The increase in website traffic could bring a slew of issues, and the company that is able to deal with the variations in traffic the most effectively will emerge.As most people have experienced a crashed site or a very slow loading time for a website when there are a lot of people using it, such as when various shopping websites may crash just before festivals as more people try to log in to the website than it was originally capable of, causing a lot of inconveniences for the users and as most people have encountered a crashed site or a very slow loading time for a website when there are a lot of people using it, it's possible that users will give the site a lower rating and instead use another site, lowering their business. As a result, a traffic management approach or plan should be implemented to limit the danger of such disasters, which could jeopardise the company's existence. Until recently, there was no need for such tools because most servers could handle the traffic influx. However, the smartphone era has increased demand to such a high level for some websites that businesses have been unable to respond quickly enough to maintain the inconsistent customer service level.

Time series analysis, a branch of statistical modeling, has emerged as a powerful tool for forecasting website traffic patterns. Among the various time series forecasting methods, the AutoRegressive Integrated Moving Average (ARIMA) model stands out as a widely adopted and reliable approach due to its ability to capture complex temporal dependencies and inherent patterns in website traffic data.

This research aims to explore the application of the ARIMA model in website traffic forecasting, investigating its effectiveness and suitability for different industry contexts. By harnessing historical website traffic data, the ARIMA model can provide valuable predictions and insights into future traffic trends, allowing businesses to anticipate fluctuations, prepare for peak demands, and make informed decisions that align with their strategic goals.

The following sections will delve into the methodology and intricacies of implementing the ARIMA model for website traffic forecasting. It will outline the data collection and pre-processing steps, detail the process of model identification and parameter estimation, and demonstrate the model's forecasting capabilities through real-world case studies. Additionally, this study will evaluate the forecasting accuracy of the ARIMA model, comparing its performance against other forecasting techniques to showcase its strengths and limitations.

With the rise of online competition and the increasing significance of digital presence, accurate website traffic forecasting becomes not merely a strategic advantage but a necessity for organizations seeking to stay ahead in the digital arena. The outcomes of this research hold immense potential for businesses, empowering them to optimize resource allocation, enhance user experiences, and build data-driven strategies to maximize their online impact.

In conclusion, the ARIMA model represents a promising avenue for website traffic forecasting, offering organizations an invaluable opportunity to proactively address user demands and tailor their online offerings. By shedding light on the intricacies of this time-tested forecasting technique, this study contributes to the body of knowledge in web analytics, equipping businesses with the tools to thrive in the ever-evolving digital landscape.

1.1 Research Questions

For the purpose of achieving good results and to solve valid problems. The following research questions were used.

1. How accurate is the ARIMA model in forecasting short-term (daily/weekly) website traffic for high-traffic e-commerce websites?

2. Can the ARIMA model effectively capture and forecast seasonal patterns in website traffic for online news portals on a monthly basis?

3. What are the optimal parameter configurations for the ARIMA model when forecasting website traffic for small business blogs with irregular patterns?

4. What are the challenges and limitations of using the ARIMA model for long-term (yearly) website traffic predictions for government informational websites?

5. Can the ARIMA model be extended to account for sudden events (e.g., product launches, viral content) and accurately predict resulting spikes in website traffic for entertainment websites?

6. How can the ARIMA model be adapted to provide probabilistic forecasts, including prediction intervals, for website traffic, considering the inherent uncertainty in such forecasts?

1.2 Objective

The main objective of website traffic forecasting using the AutoRegressive Integrated Moving Average (ARIMA) model is to provide accurate and reliable predictions of future website traffic patterns. This forecasting methodology aims to leverage historical website traffic data to identify and model underlying trends, seasonality, and temporal dependencies in the data, allowing businesses and organizations to:

1. Optimize Resource Allocation: By forecasting website traffic, businesses can efficiently allocate server resources, bandwidth, and other infrastructure to meet anticipated demands. This prevents overprovisioning, which can be costly, or underprovisioning, which may lead to performance issues during peak traffic periods.

2. Plan Marketing Strategies: Accurate traffic forecasts enable marketing teams to plan and execute targeted campaigns during periods of high traffic, ensuring that promotional efforts reach the maximum audience at the right time, leading to enhanced user engagement and conversions.

3. Improve User Experience: Anticipating fluctuations in website traffic helps organizations to provide a seamless and satisfying user experience, as they can optimize website performance to handle increased user loads without compromising on page load times and responsiveness.

4. Optimize Content Delivery: By understanding future traffic patterns, website owners can optimize the delivery of content, ensuring that popular pages and resources are readily available during peak periods, thus avoiding potential downtime or slow loading times.

5. Enhance Business Decision-Making: Accurate traffic forecasting enables organizations to make data-driven decisions related to website infrastructure investments, content creation, and overall digital strategies, leading to improved business outcomes.

6. Anticipate Seasonal Trends: ARIMA model's ability to capture seasonality patterns allows businesses to prepare for seasonal fluctuations in website traffic and align their operations and marketing efforts accordingly.

7. Evaluate Marketing and SEO Initiatives: Website traffic forecasting helps assess the effectiveness of marketing campaigns, search engine optimization(SEO) efforts, and content strategies by comparing actual traffic with predicted traffic, enabling businesses to refine their marketing tactics.

8. Respond to Unforeseen Events: Forecasting website traffic allows organizations to proactively respond to unexpected events, such as viral content or sudden spikes in user interest, by scaling up resources and adapting to changing user demands.

In conclusion, the objective of website traffic forecasting using the ARIMA model is to empower businesses with accurate predictions of future website traffic, enabling them to optimize resource allocation, plan marketing strategies, enhance user experiences, and make informed decisions that lead to a competitive advantage in the ever-evolving digital landscape.

Chapter 2

Proof of Concept

2.1 Summary of Literature Review

During the construction of the prediction model, the system successfully rebuilt the existing model and added new features, resulting in increased model efficiency. New features were used in various combinations.

1. For capturing weekly, monthly, quarterly, and yearly page popularity, use the median of specified window length in each time series as an independent feature.
2. Golden ratio-based median of medians of variable time frame windows.

To determine the importance of each feature, the study[1] analysed the obtained results and compared the accuracy's in various cases. Next, we'll try to figure out how to tweak parameters in an existing model to get better results. Study wanted to find the most suitable forecasting model based on time-series which helps us to forecast future traffic data when there is enough dataset provided.

Having this goal in mind, study began to search for models based on prediction, which would enable us to predict the value data. However, upon more research, we found that it, not a prediction but rather forecasting, after which we focused on that. Study [2] in came across so many time series forecasting models that it made our work both tedious and fun at the same time. Paper proposed a time series forecasting technique to predict internet traffic based on past values using past values. Many forecasting techniques like ARIMA are used extensively in literature for making forecasts, but it is useful mostly for a time series which is linear in nature. On the other hand, neural networks like RNN are very useful in forecasting time series which are nonlinear in nature. Proposed technique uses Discrete Wavelet Transform and uses a high pass filter and a low pass filter producing linear and nonlinear parts for the time series. The proposed technique[3] clearly outperforms ARIMA and RNN. And because of the simplicity of the technique, it can be easily employed at data centres. The paper[4] put forward a new engineering approach to prediction of campus network exit-link traffic trend. And it predicts that EPTS can have the following effect in network traffic forecasting if it has enough historical data. Web Traffic Time Series Forecasting

1. To predict network exit-link traffic trends based on historical network traffic data, we can lay out the network resource planning in advance.

2. It is easy to implement and its computing complexity is acceptable.

The paper[5] compares the traffic flow forecast effects of the LSTM network, BPNN model and ARIMA model on time series captured at a single point. The proposed LSTM network can accurately predict the traffic flow based on the relatively stable time series under normal conditions. However, the traffic system on roads is stochastic and complex, and often affected by abnormal factors like bad weather, traffic accidents and large events.

Table 2.1: Literature Review

Reference	Authors	Title	Conference
[1]	Navyasree Petluri, Eyhab Al-Masri	Web traffic prediction of Wikipedia pages	2018 IEEE International Conference on Big Data (Big Data)
[2]	Mohammmad Asifur Rahman Shuvo, et al.	Traffic forecasting using time-series analysis	2021 6th International Conference on Inventive Computation Technologies (ICICT)
[3]	Rishabh Madan, Partha Sarathi Mangipudi	Predicting computer network traffic: a time series forecasting approach using DWT, ARIMA, and RNN	2018 Eleventh International Conference on Contemporary Computing (IC3)
[4]	Fu-Ke Shen, Wei Zhang, and Pan Chang	An engineering approach to prediction of network traffic based on time-series model	2009 International Joint Conference on Artificial Intelligence
[5]	Jianhu Zheng and Mingfang Huang	Traffic flow forecast through time series analysis based on deep learning	2020, IEEE Access

Chapter 3

Proposed System

Web traffic forecasting is becoming increasingly important as businesses rely more and more on their online presence to attract and retain customers. Forecasting web traffic can help website owners and operators to optimise their website performance, allocate resources effectively, and plan for future growth. This project is implemented using the ARIMA model for web traffic forecasting.

To implement the system, a combination of Python and Google Colab was used. Python offers packages and libraries that makes implementations of machine learning algorithms and helps translate logic to code through a high level interface. It allows the researchers to focus more on the logic and less on writing code.

Google Colab is an online cloud based platform that enables researchers to have access to high power GPU's and CPU's with great capacities in which most libraries and packages are preinstalled to mitigate the time taken to find and install such packages.

ARIMA (AutoRegressive Integrated Moving Average) is a popular time series forecasting model that can be used to model and forecast time series data with trend, seasonal, and cyclic components. It is a powerful tool for modeling and forecasting web traffic because web traffic data typically exhibits these characteristics.

The proposed system will consist of the following steps:

1. Data collection: The first step in the proposed system is to collect web traffic data. We will collect data from a variety of sources, including web server

logs, Google Analytics, and other web analytics tools.

2. Data cleaning and preprocessing: Once the data is collected, we will clean and preprocess it to remove any outliers, missing values, or other errors that could affect the accuracy of the forecasting model.

3. Data visualization and analysis: After cleaning and preprocessing the data, we will use various data visualization and analysis techniques to gain insights into the patterns and trends in the web traffic data.

4. Time series modelling: Next, we will use ARIMA and SARIMA to model the web traffic data. ARIMA is a time series model that uses historical data to predict future values based on trends and patterns observed in the data.SARIMA is an extension of ARIMA model to identify seasonal patterns.

5. Model evaluation and validation: We will evaluate and validate the accuracy of the ARIMA model by comparing the predicted web traffic values to the actual values. We will use various statistical metrics, such as root mean squared error (RMSE), mean absolute error (MAE), and mean absolute percentage error (MAPE), to assess the performance of the model.

6. Forecasting: Once the model is validated, we will use it to forecast future web traffic values. The forecasted values will be used to help website owners and operators to optimize their website performance, allocate resources effectively, and plan for future growth.

3.0.1 System Architecture

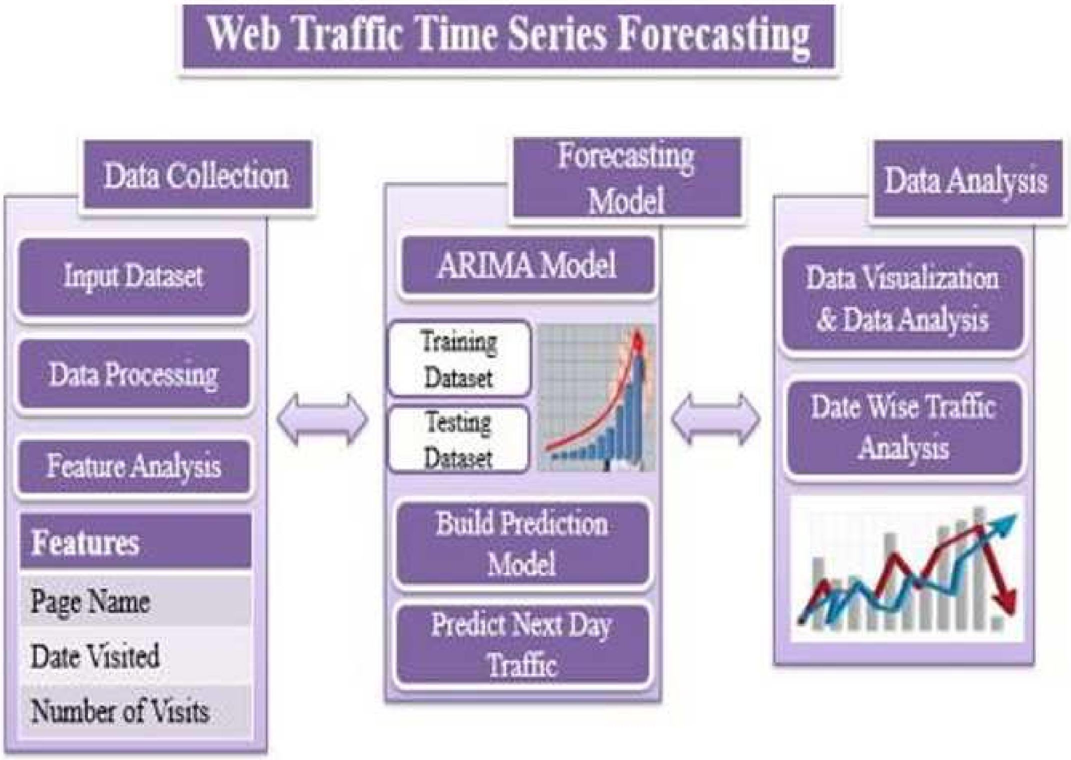


Figure 3.1: System Architecture

3.0.2 Dataset

The dataset used in this project consists of web traffic data from a GitHub repository.

Dataset Features

- This dataset has 4827 rows and 2 columns.
- The fields in the dataset are hour index and sessions
- The is of an hourly frequency collected from the website for over an year

Training and Test Data

The dataset was divided into two parts, Training set and the Validation(Test) set.

- Training Set :: 4797 Data Points
- Validation(Test) Set :: 30 Data Points

Data Preprocessing

The data is cleaned and preprocessed to remove any outliers, missing values, and other errors that could affect the accuracy of the forecasting model.

The data will be visualized and analyzed using various data visualization and analysis techniques to gain insights into the patterns and trends in the web traffic data.

Finally, the ARIMA model will be used to model the web traffic data and forecast future values.

3.1 Implementation

Website Traffic Forecasting uses statistical models and time series analysis for forecasting. The current implementation uses 2 well known statistical models.

- ARIMA - Auto Regressive Integrated Moving Average)
- SARIMA - Seasonal Auto Regressive Integrated Moving Average

3.1.1 Time Series Analysis

For as long as we have been recording data, time has been a crucial factor. In time series analysis, time is a significant variable of the data. Times series analysis helps us study our world and learn how we progress within it.

Time series analysis is a specific way of analyzing a sequence of data points collected over an interval of time. In time series analysis, analysts record data points at consistent intervals over a set period of time rather than just recording the data points intermittently or randomly. However, this type of analysis is not merely the act of collecting data over time.

What sets time series data apart from other data is that the analysis can show how variables change over time. In other words, time is a crucial variable because it shows how the data adjusts over the course of the data points as well as the final results. It provides an additional source of information and a set order of dependencies between the data.

Time series analysis typically requires a large number of data points to ensure consistency and reliability. An extensive data set ensures you have a representative sample size and that analysis can cut through noisy data. It also ensures that any trends or patterns discovered are not outliers and can account for seasonal variance. Additionally, time series data can be used for forecasting—predicting future data based on historical data.

Time series analysis helps organizations understand the underlying causes of trends or systemic patterns over time. Using data visualizations, business users can see seasonal trends and dig deeper into why these trends occur. With modern analytics platforms, these visualizations can go far beyond line graphs.

When organizations analyze data over consistent intervals, they can also use time series forecasting to predict the likelihood of future events. Time series forecasting is part of predictive analytics. It can show likely changes in the data, like seasonality or cyclic behavior, which provides a better understanding of data variables and helps forecast better.

Time series analysis is used for non-stationary data—things that are constantly fluctuating over time or are affected by time. Industries like finance, retail, and economics frequently use time series analysis because currency and sales are always changing.

Stock market analysis is an excellent example of time series analysis in action, especially with automated trading algorithms. Likewise, time series analysis is ideal for forecasting weather changes, helping meteorologists predict everything from tomorrow's weather report to future years of climate change.

Examples of time series analysis in action include:

- Weather data
- Rainfall measurements
- Temperature readings
- Heart Rate Monitoring(ECG)
- Brain monitoring (EEG)
- Quarterly sales
- Stock prices
- Automated stock trading
- Industry forecast

Time Series Analysis Type

Models of time series analysis include:

1. Classification: Identifies and assigns categories to the data.
2. Curve fitting: Plots the data along a curve to study the relationships of variables within the data.
3. Descriptive analysis: Identifies patterns in time series data, like trends, cycles, or seasonal variation.
4. Explanative analysis: Attempts to understand the data and the relationships within it, as well as cause and effect.
5. Exploratory analysis: Highlights the main characteristics of the time series data, usually in a visual format.
6. Forecasting: Predicts future data. This type is based on historical trends. It uses the historical data as a model for future data, predicting scenarios that could happen along future plot points.
7. Intervention analysis: Studies how an event can change the data.
8. Segmentation: Splits the data into segments to show the underlying properties of the source information.

Data Classification

Further, time series data can be classified into two main categories:

1. Stock time series data means measuring attributes at a certain point in time, like a static snapshot of the information as it was.
2. Flow time series data means measuring the activity of the attributes over a certain period, which is generally part of the total whole and makes up a portion of the results.

Data Variations

In time series data, variations can occur sporadically throughout the data:

1. Functional analysis can pick out the patterns and relationships within the data to identify notable events.
2. Trend analysis means determining consistent movement in a certain direction. There are two types of trends: deterministic, where we can find the underlying cause, and stochastic, which is random and inexplicable.
3. Seasonal variation describes events that occur at specific and regular intervals during the course of a year. Serial dependence occurs when data points close together in time tend to be related.
4. Time series analysis and forecasting models must define the types of data relevant to answering the business question. Once analysts have chosen the relevant data they want to analyze, they choose what types of analysis and techniques are the best fit.

Important Considerations of Time Series Analysis

While time series data is data collected over time, there are different types of data that describe how and when that time data was recorded. For example:

1. Time series data is data that is recorded over consistent intervals of time.
2. Cross-sectional data consists of several variables recorded at the same time.
3. Pooled data is a combination of both time series data and cross-sectional data.

Time Series Models and Techniques

Just as there are many types and models, there are also a variety of methods to study data. Here are the three most common.

1. Box-Jenkins ARIMA models: These univariate models are used to better understand a single time-dependent variable, such as temperature over time, and to predict future data points of variables. These models work on the assumption that the data is stationary.

   Analysts have to account for and remove as many differences and seasonalities in past data points as they can. Thankfully, the ARIMA model includes terms to account for moving averages, seasonal difference operators, and autoregressive terms within the model.

2. Box-Jenkins Multivariate Models: Multivariate models are used to analyze more than one time-dependent variable, such as temperature and humidity, over time.

3. Holt-Winters Method: The Holt-Winters method is an exponential smoothing technique. It is designed to predict outcomes, provided that the data points include seasonality.

3.1.2 ARIMA(Auto Regressive Integrated Moving Average

ARIMA (Auto regressive Integrated Moving Average model) is a statistical analysis technique that uses time series data to better understand or forecast future trends. An autoregressive integrated moving average model is a type of regression analysis that determines how strong one dependent variable is in comparison to other changing variables.

The purpose of the model is to anticipate future securities or financial market movements by looking at the discrepancies between values in a series rather than actual values.

Any ‘non-seasonal’ time series that exhibits patterns and is not a random white noise can be modeled with ARIMA models.

An ARIMA model is characterized by 3 terms: p, d, q, where,

- p is the order of the AR term.
- q is the order of the MA term.
- d is the number of differencing required to make the time series stationary.

If a time series, has seasonal patterns, then you need to add seasonal terms and it becomes SARIMA, short for ‘Seasonal ARIMA’.

The first step to build an ARIMA model is to make the time series stationary. Because, term ‘Auto Regressive’ in ARIMA means it is a linear regression model that uses its own lags as predictors. Linear regression models work best when the predictors are not correlated and are independent of each other.

So how to make a series stationary? The most common approach is to difference it. That is, subtract the previous value from the current value. Sometimes, depending on the complexity of the series, more than one differencing may be needed.

The value of d, therefore, is the minimum number of differencing needed to make the series stationary. And if the time series is already stationary, then d = 0.

Next, what are the ‘p’ and ‘q’ terms?

‘p’ is the order of the ‘Auto Regressive’ (AR) term. It refers to the number of lags of Y to be used as predictors. And ‘q’ is the order of the ‘Moving Average’ (MA) term. It refers to the number of lagged forecast errors that should go into the ARIMA Model.

A pure Auto Regressive (AR only) model is one where Yt depends only on its own lags. That is, Yt is a function of the ‘lags of Yt’.

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_p Y_{t-p} + \epsilon_1 \tag{3.1}$$

where, $Yt - 1$ is the lag1 of the series, $\beta 1$ is the coefficient of lag1 that the model estimates and α is the intercept term, also estimated by the model.

Likewise a pure Moving Average (MA only) model is one where Yt depends only on the lagged forecast errors.

$$Y_t = \alpha + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + .. + \phi_q \epsilon_{t-q} \tag{3.2}$$

where the error terms are the errors of the autoregressive models of the respective lags. The errors Et and E(t-1) are the errors from the following equations,

$$Y_t = \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_0 Y_0 + \epsilon_t \tag{3.3}$$

$$Y_{t-1} = \beta_1 Y_{t-2} + \beta_2 Y_{t-3} + .. + \beta_0 Y_0 + \epsilon_{t-1} \tag{3.4}$$

That was AR and MA models respectively.

An ARIMA model is one where the time series was differenced at least once to make it stationary and you combine the AR and the MA terms. So the equation becomes:

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_p Y_{t-p} \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + .. + \phi_q \epsilon_{t-q} \tag{3.5}$$

ARIMA model in words:

Predicted Yt = Constant + Linear combination Lags of Y (upto p lags) + Linear Combination of Lagged forecast errors (upto q lags)

The objective, therefore, is to identify the values of p, d and q.

How Does ARIMA Forecasting Work?

ARIMA forecasting is achieved by plugging in time series data for the variable of interest. Statistical software will identify the appropriate number of lags or amount of differencing to be applied to the data and check for stationarity. It will then output the results, which are often interpreted similarly to that of a multiple linear regression mode.

The Bottom Line

The ARIMA model is used as a forecasting tool to predict how something will act in the future based on past performance. It is used in technical analysis to predict an asset's future performance.

ARIMA modeling is generally inadequate for long-term forecastings, such as more than six months ahead, because it uses past data and parameters that are influenced by human thinking. For this reason, it is best used with other technical analysis tools to get a clearer picture of an asset's performance.The ARIMA model is an incredibly valuable tool for aspiring data scientists.

3.1.3 SARIMA(Seasonal Auto Regressive Integrated Moving Average)

Time series data, characterized by observations over successive time intervals, is ubiquitous across various domains such as finance, economics, weather, and more. Accurate forecasting of future values within these datasets is essential for decision-making, planning, and resource allocation. The Seasonal Autoregressive Integrated Moving Average (SARIMA) model emerges as a powerful tool in time series analysis, building upon the foundation of ARIMA models while incorporating the ability to handle seasonality.

The SARIMA model addresses the limitations of ARIMA by introducing additional seasonal components. This extension equips the model with the ability to account for recurring patterns that occur at regular intervals, enhancing its forecasting accuracy for datasets exhibiting both non-seasonal and seasonal variations.

SARIMA stands for ”Seasonal AutoRegressive Integrated Moving Average.” It is an extension of the ARIMA (AutoRegressive Integrated Moving Average) model that takes into account the seasonality present in a time series data. SARIMA is used for time series forecasting and is particularly effective when the data exhibits seasonality, which refers to repeating patterns or trends that occur at regular intervals.

In SARIMA, similar to ARIMA, the model includes three main components:

- AutoRegressive (AR) terms: These terms represent the relationship between the current value and its past values in a time series.
- Integrated (I) terms: These terms capture the differencing needed to make the time series stationary, i.e., removing the trend and making the series mean-stationary.
- Moving Average (MA) terms: These terms model the relationship between the current value and past forecast errors (residuals) in a time series.

Additionally, SARIMA introduces the notion of seasonality. It includes seasonal AR, I, and MA terms, denoted as SAR, SI, and SMA terms, respectively. These seasonal terms capture the influence of previous seasonal values, seasonal differences, and seasonal forecast errors.

SARIMA is defined by its notation, SARIMA(p, d, q)(P, D, Q, s), where 'p', 'd', and 'q' represent the non-seasonal autoregressive, differencing, and moving average orders, respectively. The seasonal component is represented by 'P', 'D', and 'Q', capturing the seasonal autoregressive, differencing, and moving average orders, respectively. The parameter 's' indicates the length of the seasonal cycle.

The construction of a SARIMA model involves several key steps:

- Data Preprocessing: Transforming the raw time series data to ensure stationarity through differencing.

- Order Selection: Determining the appropriate values of 'p', 'd', 'q', 'P', 'D', and 'Q' based on techniques like autocorrelation and partial autocorrelation functions.

- Model Fitting: Estimating the model parameters using optimization algorithms to minimize forecast errors.

- Seasonal Differencing: Applying seasonal differencing to account for the seasonal component.

- Forecasting: Utilizing the fitted SARIMA model to make future predictions by incorporating past observations and forecast errors.

SARIMA models offer several advantages, including their ability to handle both non-seasonal and seasonal patterns, resulting in accurate forecasts for a wide range of time series data. However, SARIMA models can be complex to configure due to the multiple parameters involved, requiring careful tuning. Moreover, SARIMA may not be suitable for datasets with irregular or non-repetitive seasonal patterns. SARIMA finds applications in various fields, such as:

- Economic Forecasting: Predicting economic indicators like GDP, inflation rates, and stock prices.

- Demand Forecasting: Estimating product demand for inventory management and supply chain optimization.

- Meteorology: Forecasting weather variables like temperature, precipitation, and wind speed.

- Energy Consumption: Predicting energy usage for efficient resource allocation.

In summary, SARIMA is a powerful tool for modeling and forecasting time series data with both regular and seasonal patterns. It provides a way to capture and account for the inherent seasonality in the data, making it particularly useful for applications where the underlying patterns follow seasonal trends.In the realm of time series analysis, SARIMA stands as a robust methodology that bridges the gap between ARIMA and seasonal patterns.

Its ability to capture both non-seasonal and seasonal behaviors makes it a valuable tool for accurate forecasting, driving informed decision-making across diverse industries. However, successful implementation requires a comprehensive understanding of its components and careful parameter selection to harness its predictive potential effectively.

3.2 Code Explanation

1. Importing Packages: Various Python packages such as Pandas, NumPy, Matplotlib, and Statsmodels are imported to handle data manipulation, visualization, and time series analysis.

2. Functions for Modularization: The code begins by defining modular functions for different tasks. These functions improve code organization, reusability, and debugging.

   Functions are defined for loading data, decomposing time series, performing ARIMA/SARIMA forecasting, and calculating forecast accuracy.

3. Load Web Traffic Data: The web traffic data is loaded from a CSV file using the ‘load data‘ function. The data is converted into a Pandas DataFrame with a DatetimeIndex, making it suitable for time series analysis.

4. Plot Web Traffic Data: The loaded web traffic data is visualized using Matplotlib. A line plot is created to show the ’Sessions’ column against the DatetimeIndex, giving an overview of the web traffic trends.

5. Decompose Time Series: The time series data is decomposed into trend, seasonal, and residual components using the ‘decompose_series‘ function from Statsmodels.

   This decomposition helps identify underlying patterns and irregularities in the data.

6. Autocorrelation and Partial Autocorrelation Plots: Autocorrelation and partial autocorrelation plots are created to analyze the correlation of the time

series with its lagged values. These plots help determine the appropriate parameters for the ARIMA model.

7. ARIMA and SARIMA Forecasting: An ARIMA (AutoRegressive Integrated Moving Average) model is defined using the 'arima forecast' function. The model is fitted to the training data, and a forecast is generated for the next 30 periods. The ARIMA order and seasonal order parameters are specified.

8. Results Visualization: The forecasted values are plotted along with the historical training data and the test data. This visualization helps assess the performance of the ARIMA model and understand its accuracy.

9. Residual Analysis: Residuals (the differences between actual and predicted values) are plotted to analyze the model's prediction errors. The zero line is included for reference.

10. Actual vs. Predicted Plot: The actual test data and the predicted values are plotted together to visually compare the model's predictions with the real observations.

11. Forecast Accuracy Calculation: The Mean Absolute Percentage Error (MAPE) is calculated to quantify the accuracy of the ARIMA model's forecasts for the last 30 values.

    The code demonstrates the process of web traffic forecasting using the ARIMA model. It includes data loading, visualization, decomposition, parameter selection, forecasting, and accuracy evaluation. The modular approach enhances code readability, maintainability, and debugging.

Chapter 4

Result and Analysis

The accuracy of the developed ARIMA and SARIMA models were done using the segmented test data. As observed from the graph, the system is able to accurately predict the values into the future. Further into the future, the system is able to capture the trend but not the exact values.

It is also noted that the accuracy of the system decreases as we move into the future. To mitigate this problem, website for casting should be done with live data and the predictions should be constantly updated.

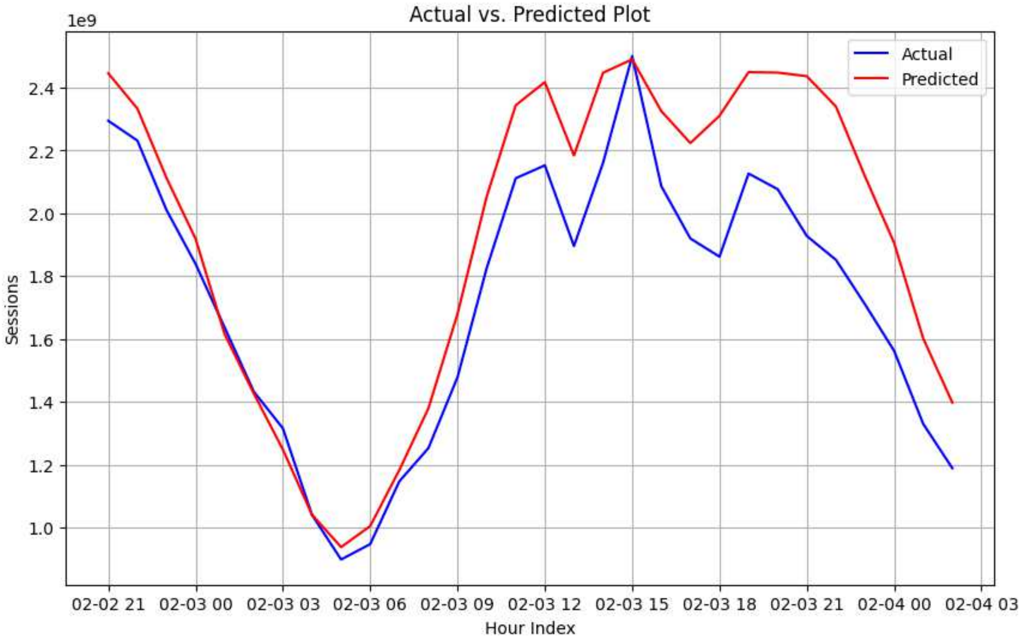


Figure 4.1: Actual vs. Predicted Output

ARIMA Model Error (MAPE) for the last test data of 30 values is 11.67% The lower the value of the Mean Absolute Percentage Error, the better the prediction of the model.

4.1 Advantages

- Time Series Analysis: ARIMA is specifically designed for time series data, making it well-suited for modeling and forecasting website traffic, which often exhibits temporal patterns and trends.
- Accounting for Seasonality: ARIMA can handle seasonality, capturing daily, weekly, or monthly patterns in website traffic, which is common for many online platforms.
- Data-Driven Insights: ARIMA provides insights into historical traffic patterns, helping website owners and administrators understand when traffic peaks and troughs occur.
- Short-Term Predictions: ARIMA excels at short-term forecasting, making it suitable for predicting website traffic over the immediate future, such as the next few hours or days.
- Interpretability: The ARIMA model's parameters have clear interpretations, allowing users to understand the impact of autoregressive and moving average components on the forecast.
- Relatively Simple: ARIMA is a well-established and relatively simple time series forecasting method, making it accessible to analysts and data scientists with varying levels of expertise.
- Quick Implementation: Once the model is set up and tuned, generating forecasts is relatively fast, making it suitable for real-time or near-real-time monitoring and decision-making.

- Baseline Model: ARIMA can serve as a baseline model for website traffic forecasting, providing a solid foundation for more complex and advanced models.

- Model Diagnostic Tools: ARIMA offers tools to diagnose the quality of the model fit, including residual analysis, ACF (AutoCorrelation Function), and PACF (Partial AutoCorrelation Function) plots.

- Adaptability: ARIMA can be adapted to handle additional external factors that might impact website traffic, such as holidays, promotions, or special events.

- Predictive Analytics: Accurate website traffic forecasts from ARIMA enable better resource allocation, capacity planning, and optimization of server resources.

- Historical Comparison: Forecasted traffic can be compared with actual observed traffic, enabling businesses to assess the accuracy of their predictions and improve their forecasting process over time.

While ARIMA has its advantages, it's important to note that its effectiveness may vary depending on the specific characteristics of the website traffic data. For more complex patterns, advanced forecasting techniques or machine learning models may be necessary.

4.2 Limitations

- Linear Assumption: ARIMA assumes a linear relationship between past and future values, which might not capture complex and nonlinear patterns in website traffic data.

- Limited Handling of External Factors: ARIMA is primarily designed for time series data and may struggle to incorporate external factors such as marketing campaigns, holidays, or sudden events that can significantly impact website traffic.

- Data Quality and Preprocessing: ARIMA is sensitive to outliers, missing values, and noise in the data. Cleaning and preprocessing the data becomes crucial for accurate forecasts.

- Lag Selection: Choosing appropriate lags for autoregressive and moving average terms can be challenging and may require domain knowledge or trial-and-error.

- Seasonal Patterns: ARIMA may struggle with capturing irregular or complex seasonal patterns in website traffic data, particularly if the seasonality is not consistent.

- Long-Term Predictions: ARIMA is better suited for short-term forecasts and may not perform well when trying to predict website traffic far into the future.

- Model Complexity: While ARIMA is relatively simple, it may struggle to capture intricate patterns present in high-frequency website traffic data.

- Changing Trends: If website traffic patterns undergo significant changes over time, ARIMA may not adapt well to abrupt shifts or evolving trends.

- Overfitting and Underfitting: Finding the right balance between model complexity and accuracy can be tricky, leading to potential issues of overfitting or underfitting the data.

- Limited Forecast Uncertainty: ARIMA provides point forecasts but does not inherently provide measures of uncertainty or confidence intervals for its predictions.

- Parameter Sensitivity: The performance of the ARIMA model can be sensitive to the values of its hyperparameters, and different combinations may lead to varying results.

- High-Frequency Data: For websites with very high-frequency traffic data (e.g., data collected in seconds), ARIMA may not be the most suitable choice due to the computational complexity and the assumption of stationarity.

- Model Interpretability: While ARIMA's parameters have clear interpretations, the model's inner workings might not provide deeper insights into the underlying dynamics of website traffic.

- Complex Seasonality: If the website traffic exhibits complex seasonality that is not easily captured by ARIMA's seasonal component, the model's accuracy may be compromised.

- Data Volume: ARIMA may require a sufficient amount of historical data for accurate forecasting, which can be a limitation for new websites or those with limited data.

Given these limitations, it's important to assess whether the characteristics of the website traffic data align with ARIMA's assumptions and capabilities before using this model for forecasting. In cases where the data exhibits complex patterns or is influenced by various external factors, more advanced forecasting techniques or machine learning models may be more suitable.

4.3 Future Scope

The future scope of website traffic forecasting using the ARIMA (AutoRegressive Integrated Moving Average) model includes several potential avenues for improvement and expansion:

- Enhanced Model Variants: Researchers and practitioners can develop more advanced variants of the ARIMA model that address its limitations. This could involve incorporating additional components or modifying the model structure to better capture complex patterns in website traffic data.

- Hybrid Models: Integrating the ARIMA model with other forecasting methods, such as machine learning techniques or neural networks, can potentially improve accuracy by leveraging the strengths of different approaches.

- Incorporating External Factors: Future research can focus on methods to effectively incorporate external factors, such as social media trends, marketing campaigns, or economic indicators, into the ARIMA model to enhance its predictive capabilities.

- Long-Term Forecasting: Extending the ARIMA model to handle long-term forecasting by addressing issues related to forecasting horizon and trend shifts will broaden its applicability.

- Real-Time Forecasting: Developing real-time forecasting capabilities using ARIMA models can be valuable for website operators to make timely decisions based on up-to-date predictions.

An advanced system for website traffic forecasting utilizes cutting-edge techniques like LSTM neural networks, Prophet, and ensemble methods. Unlike ARIMA, these models capture complex temporal patterns and non-linear relationships, adapting to changing trends and anomalies in data. LSTM excels in learning sequential dependencies, while Prophet integrates seasonality and special events. Ensemble methods enhance accuracy and robustness. These systems offer automatic feature extraction, hyperparameter tuning, and uncertainty estimation, providing actionable insights for optimizing online platforms and user experiences.

Overall, the future scope of website traffic forecasting using the ARIMA model involves advancing its capabilities, addressing limitations, and exploring innovative ways to enhance its accuracy, usability, and applicability in the dynamic context of web traffic prediction.

Chapter 5

Conclusion

This project aims to develop a reliable forecasting model for predicting the future traffic of websites. The proposed approach involved using the ARIMA model on the Web Traffic Time Series Forecasting dataset which aims to successfully train a model using features such as page name, visited date, and the number of page visits to predict future web traffic.In conclusion, website traffic forecasting using the ARIMA model presents a valuable approach for predicting future user activity. ARIMA leverages time-series analysis to capture underlying patterns, trends, and seasonality within web traffic data. Its simplicity and interpretability make it a useful tool for short to medium-term predictions. By decomposing the data, identifying autocorrelation, and incorporating exogenous variables, ARIMA enhances accuracy. However, it may struggle with complex nonlinear relationships and abrupt changes. Despite its limitations, ARIMA remains a foundational method in web traffic forecasting, providing valuable insights for businesses to optimize resource allocation, enhance user experiences, and make informed decisions.

By using this model, website owners and developers can gain valuable insights into their website's traffic patterns and make informed decisions about how to optimize their website's performance.

References

[1] Navyasree Petluri and Eyhab Al-Masri. Web traffic prediction of wikipedia pages. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 5427–5429. IEEE, 2018.

[2] Mohammmad Asifur Rahman Shuvo, Muhtadi Zubair, Afsara Tahsin Purnota, Sarowar Hossain, and Muhammad Iqbal Hossain. Traffic forecasting using time-series analysis. In *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, pages 269–274. IEEE, 2021.

[3] Rishabh Madan and Partha Sarathi Mangipudi. Predicting computer network traffic: a time series forecasting approach using dwt, arima and rnn. In *2018 Eleventh International Conference on Contemporary Computing (IC3)*, pages 1–5. IEEE, 2018.

[4] Fu-Ke Shen, Wei Zhang, and Pan Chang. An engineering approach to prediction of network traffic based on time-series model. In *2009 International Joint Conference on Artificial Intelligence*, pages 432–435. IEEE, 2009.

[5] Jianhu Zheng and Mingfang Huang. Traffic flow forecast through time series analysis based on deep learning. *IEEE Access*, 8:82562–82570, 2020.

[6] Pablo Montero-Manso, George Athanasopoulos, Rob J Hyndman, and Thiyanga S Talagala. Fforma: Feature-based forecast model averaging. *International Journal of Forecasting*, 36(1):86–92, 2020.

[7] Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. *Advances in neural information processing systems*, 31, 2018.

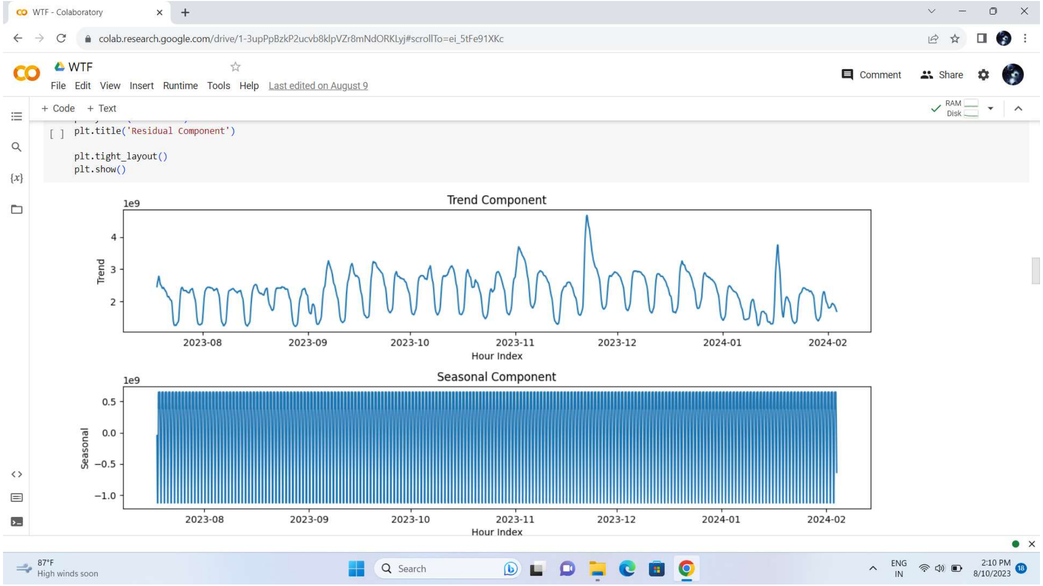
[8] Ahmed Tealab. Time series forecasting using artificial neural networks methodologies: A systematic review. *Future Computing and Informatics Journal*, 3(2):334–340, 2018.

[9] Hristos Tyralis and Georgia Papacharalampous. Variable selection in time series forecasting using random forests. *Algorithms*, 10(4):114, 2017.

[10] Wei-Chih Chen, Wen-Hui Chen, and Sheng-Yuan Yang. A big data and time series analysis technology-based multi-agent system for smart tourism. *Applied Sciences*, 8(6):947, 2018.

Appendix

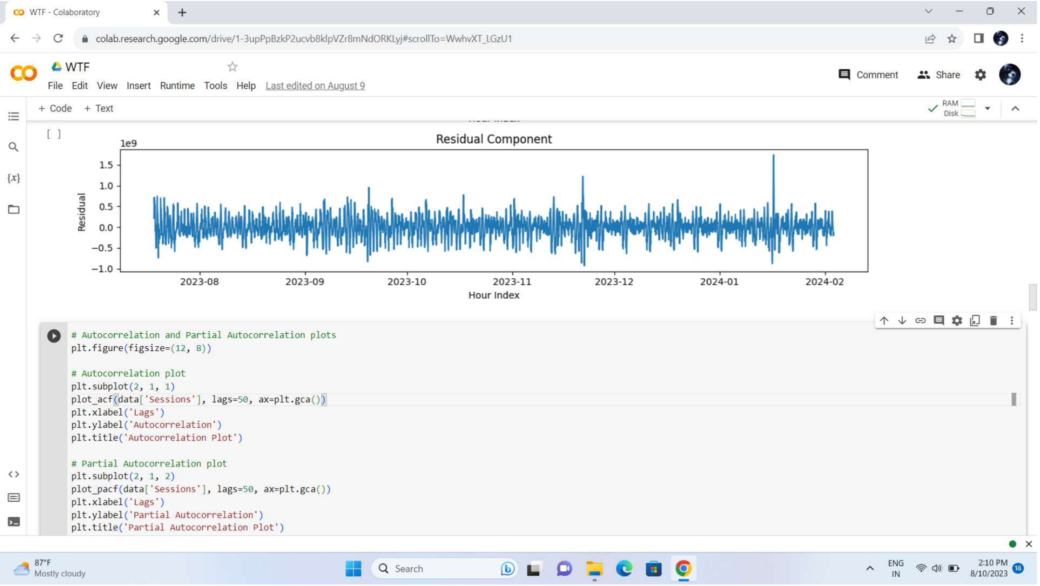
.1 Appendix A-Screenshot of the project



```
[ ] plt.title('Residual Component')

    plt.tight_layout()
    plt.show()
```

Figure 1: Plot of Trend Component and Seasonal Component

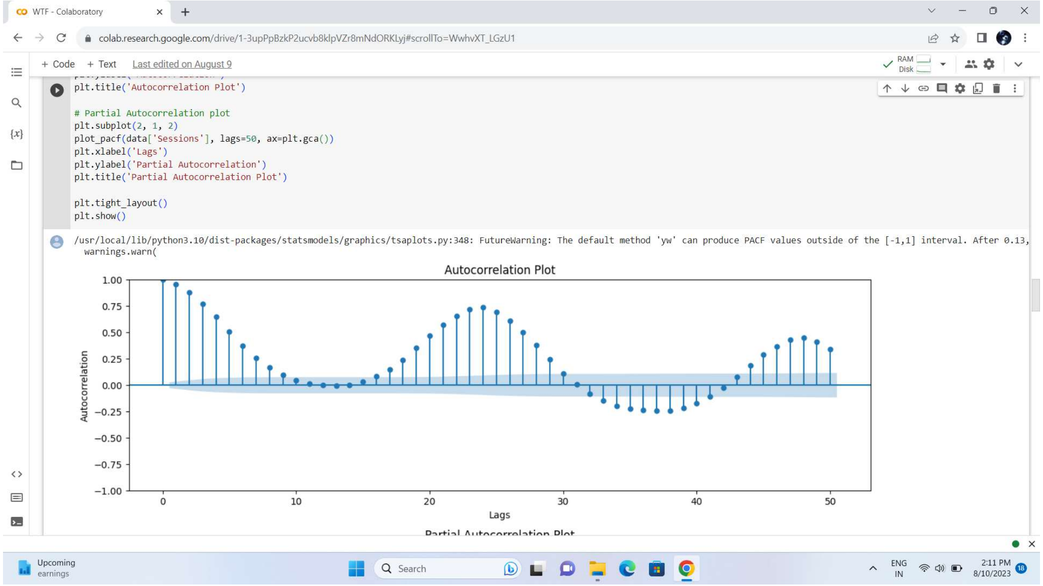


```
# Autocorrelation and Partial Autocorrelation plots
plt.figure(figsize=(12, 8))

# Autocorrelation plot
plt.subplot(2, 1, 1)
plot_acf(data['Sessions'], lags=50, ax=plt.gca())
plt.xlabel('Lags')
plt.ylabel('Autocorrelation')
plt.title('Autocorrelation Plot')

# Partial Autocorrelation plot
plt.subplot(2, 1, 2)
plot_pacf(data['Sessions'], lags=50, ax=plt.gca())
plt.xlabel('Lags')
plt.ylabel('Partial Autocorrelation')
plt.title('Partial Autocorrelation Plot')
```

Figure 2: Plot of Residual Component



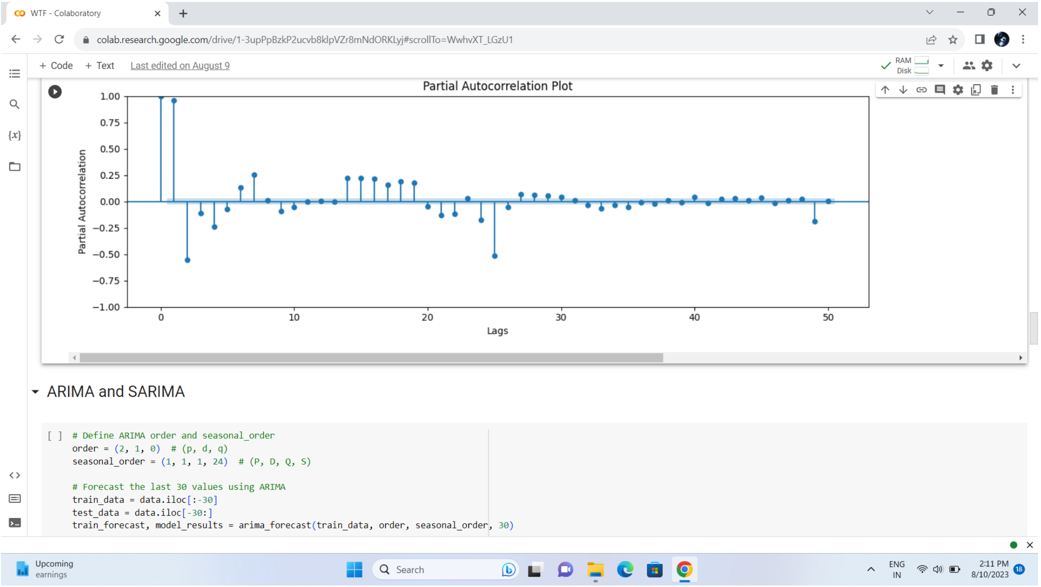
```
plt.title('Autocorrelation Plot')

# Partial Autocorrelation plot
plt.subplot(2, 1, 2)
plot_pacf(data['Sessions'], lags=50, ax=plt.gca())
plt.xlabel('Lags')
plt.ylabel('Partial Autocorrelation')
plt.title('Partial Autocorrelation Plot')

plt.tight_layout()
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/statsmodels/graphics/tsaplots.py:348: FutureWarning: The default method 'yw' can produce PACF values outside of the [-1,1] interval. After 0.13,
  warnings.warn(
```

Figure 3: Plot of Autocorrelation

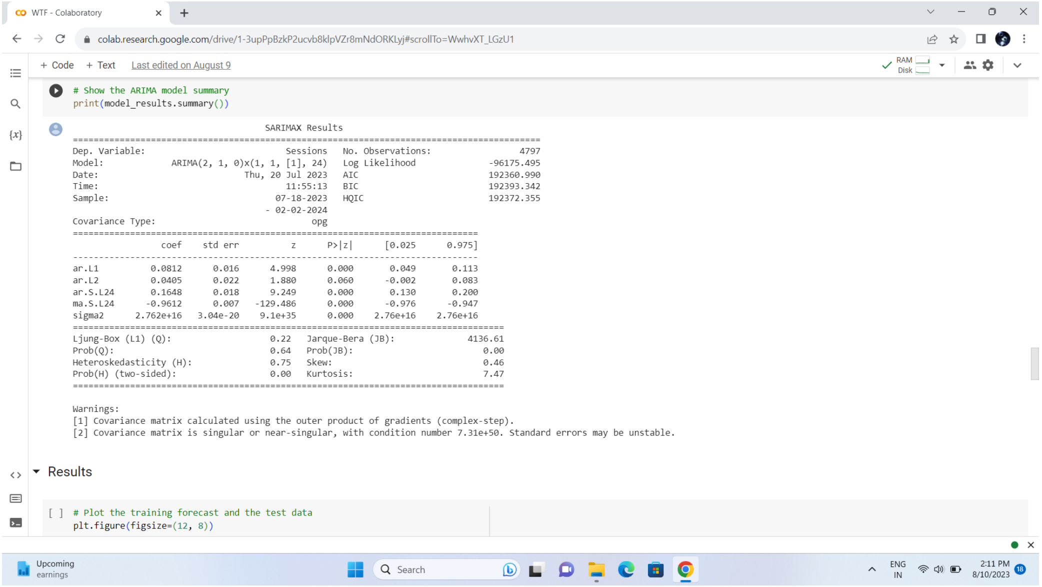


ARIMA and SARIMA

```
# Define ARIMA order and seasonal_order
order = (2, 1, 0)  # (p, d, q)
seasonal_order = (1, 1, 1, 24)  # (P, D, Q, S)

# Forecast the last 30 values using ARIMA
train_data = data.iloc[:-30]
test_data = data.iloc[-30:]
train_forecast, model_results = arima_forecast(train_data, order, seasonal_order, 30)
```

Figure 4: Plot of Partial Autocorrelation



```
# Show the ARIMA model summary
print(model_results.summary())
```

```
                                     SARIMAX Results                                      
==========================================================================================
Dep. Variable:                           Sessions   No. Observations:                 4797
Model:             ARIMA(2, 1, 0)x(1, 1, [1], 24)   Log Likelihood              -96175.495
Date:                            Thu, 20 Jul 2023   AIC                         192360.990
Time:                                    11:55:13   BIC                         192393.342
Sample:                                07-18-2023   HQIC                        192372.355
                                     - 02-02-2024                                         
Covariance Type:                              opg                                         
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.0812      0.016      4.998      0.000       0.049       0.113
ar.L2          0.0405      0.022      1.880      0.060      -0.002       0.083
ar.S.L24       0.1648      0.018      9.249      0.000       0.130       0.200
ma.S.L24      -0.9612      0.007   -129.486      0.000      -0.976      -0.947
sigma2      2.762e+16   3.04e-20    9.1e+35      0.000    2.76e+16    2.76e+16
===================================================================================
Ljung-Box (L1) (Q):                   0.22   Jarque-Bera (JB):              4136.61
Prob(Q):                              0.64   Prob(JB):                         0.00
Heteroskedasticity (H):               0.75   Skew:                             0.46
Prob(H) (two-sided):                  0.00   Kurtosis:                         7.47
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
[2] Covariance matrix is singular or near-singular, with condition number 7.31e+50. Standard errors may be unstable.
```

Results

```
# Plot the training forecast and the test data
plt.figure(figsize=(12, 8))
```

Figure 5: ARIMA model summary

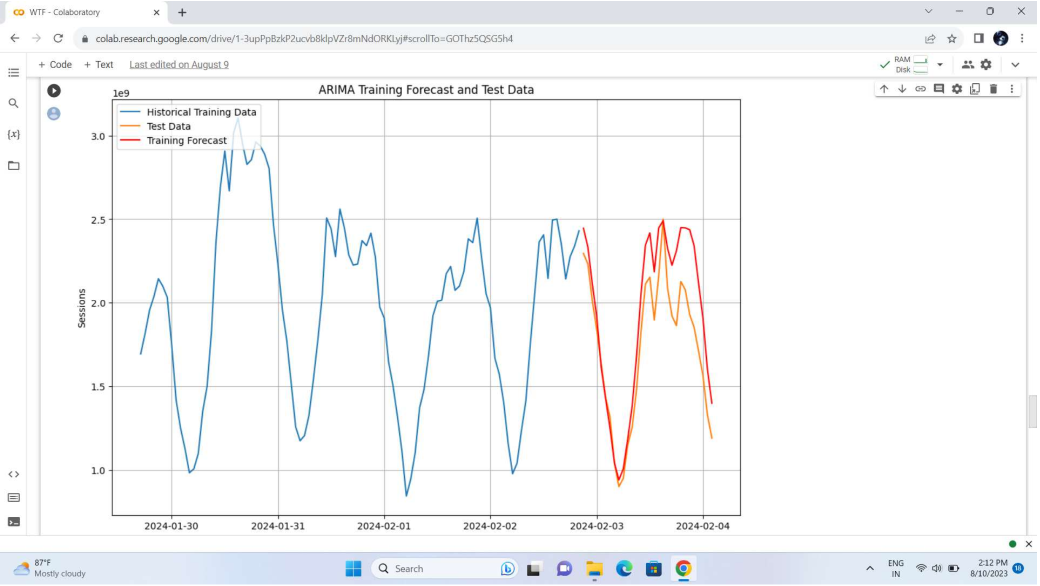
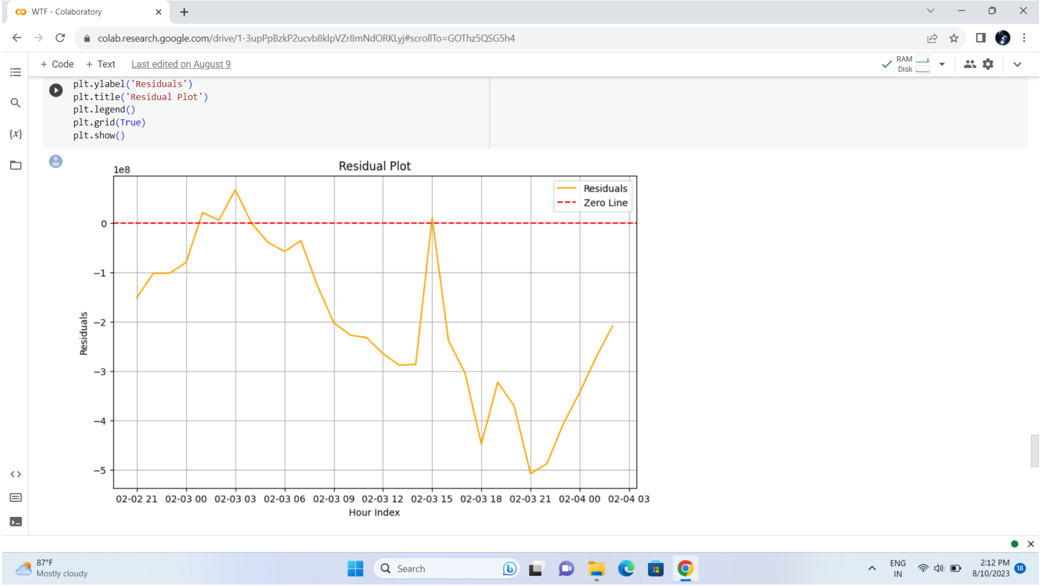


Figure 6: Plot of ARIMA training model and test data



```
plt.ylabel('Residuals')
plt.title('Residual Plot')
plt.legend()
plt.grid(True)
plt.show()
```

Figure 7: Residual Plot

.2 Appendix B-Sample Code

```
# Importing necessary packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

# Functions for modularizing the code

def load_data(csv_file):
    # Load web traffic data from a CSV file
    data = pd.read_csv(csv_file)
    data['Hour␣Index'] = pd.to_datetime('2023-07-18') + pd.to_timedelta(data['Hour␣Index'], unit='h')
    data.set_index('Hour␣Index', inplace=True)
    return data

def decompose_series(data):
    # Decompose the time series to check for seasonality
    decomposition = sm.tsa.seasonal_decompose(data['Sessions'], model='additive')
    trend = decomposition.trend
```

```python
    seasonal = decomposition.seasonal
    residual = decomposition.resid
    return trend, seasonal, residual

def arima_forecast(data, order, seasonal_order=None, forecast_periods=30):
    # Perform ARIMA/SARIMA forecasting
    model = sm.tsa.ARIMA(data['Sessions'], order=order, seasonal_order=seasonal_order, freq='H')
    results = model.fit()
    forecast = results.forecast(steps=forecast_periods)
    return forecast, results

def calculate_accuracy(test_data, forecast):
    # Calculate forecast accuracy using Mean Absolute Percentage Error (MAPE)
    abs_percentage_errors = np.abs((test_data - forecast) / test_data)
    mape = np.mean(abs_percentage_errors) * 100
    return mape

# Web-Traffic Data
data = load_data('Web_Data.csv')
print(data)

```

```
    # Plot the web traffic data
    plt.figure(figsize=(10, 6))
    plt.plot(data.index, data[’Sessions’], label=’Web␣Traffic’)
    plt.xlabel(’Hour␣Index’)
    plt.ylabel(’Sessions’)
    plt.title(’Web␣Traffic␣Data’)
    plt.legend(loc=’upper␣left’)
    plt.grid(True)
    plt.show()

    # Analysis
    # Decompose the time series to check for seasonality
    trend, seasonal, residual = decompose_series(data)

    # Plot the decomposed components
    plt.figure(figsize=(12, 8))

    # Plot the Trend component
    plt.subplot(3, 1, 1)
    plt.plot(trend)
    plt.xlabel(’Hour␣Index’)
    plt.ylabel(’Trend’)
    plt.title(’Trend␣Component’)

    # Plot the Seasonal component
```

```
plt.subplot(3, 1, 2)
plt.plot(seasonal)
plt.xlabel(’Hour␣Index’)
plt.ylabel(’Seasonal’)
plt.title(’Seasonal␣Component’)

# Plot the Residual component
plt.subplot(3, 1, 3)
plt.plot(residual)
plt.xlabel(’Hour␣Index’)
plt.ylabel(’Residual’)
plt.title(’Residual␣Component’)

plt.tight_layout()
plt.show()

# Autocorrelation and Partial Autocorrelation plots
plt.figure(figsize=(12, 8))

# Autocorrelation plot
plt.subplot(2, 1, 1)
plot_acf(data[’Sessions’], lags=50, ax=plt.gca())
plt.xlabel(’Lags’)
plt.ylabel(’Autocorrelation’)
plt.title(’Autocorrelation␣Plot’)

```

```
# Partial Autocorrelation plot
plt.subplot(2, 1, 2)
plot_pacf(data[’Sessions’], lags=50, ax=plt.gca())
plt.xlabel(’Lags’)
plt.ylabel(’Partial␣Autocorrelation’)
plt.title(’Partial␣Autocorrelation␣Plot’)

plt.tight_layout()
plt.show()

## ARIMA and SARIMA

# Define ARIMA order and seasonal_order
order = (2, 1, 0)  # (p, d, q)
seasonal_order = (1, 1, 1, 24)  # (P, D, Q, S)

# Forecast the last 30 values using ARIMA
train_data = data.iloc[:-30]
test_data = data.iloc[-30:]
train_forecast, model_results =
   arima_forecast(train_data, order, seasonal_order,
   30)

# Show the ARIMA model summary
print(model_results.summary())

```

```
## Results

# Plot the training forecast and the test data
plt.figure(figsize=(12, 8))

# Plot the historical training data
plt.plot(train_data.index[-100:], train_data[’Sessions’].tail(100), label=’Historical␣Training␣Data’)

# Plot the test data
plt.plot(test_data.index, test_data[’Sessions’], label=’Test␣Data’)

# Plot the forecast for the test data
plt.plot(test_data.index, train_forecast, label=’Training␣Forecast’, color=’red’)

# Set labels and title
plt.xlabel(’Hour␣Index’)
plt.ylabel(’Sessions’)
plt.title(’ARIMA␣Training␣Forecast␣and␣Test␣Data’)
plt.legend(loc=’upper␣left’)
plt.grid(True)  # Add grid lines for better readability
plt.show()

```

```
    # Residual plot
    plt.figure(figsize=(10, 6))
    plt.plot(test_data.index, test_data[’Sessions’] -
       train_forecast, label=’Residuals’, color=’orange’)
    plt.axhline(y=0, color=’red’, linestyle=’--’,
       label=’Zero␣Line’)
    plt.xlabel(’Hour␣Index’)
    plt.ylabel(’Residuals’)
    plt.title(’Residual␣Plot’)
    plt.legend()
    plt.grid(True)
    plt.show()

    # Actual vs. Predicted Plot
    plt.figure(figsize=(10, 6))
    plt.plot(test_data.index, test_data[’Sessions’],
       label=’Actual’, color=’blue’)
    plt.plot(test_data.index, train_forecast,
       label=’Predicted’, color=’red’)
    plt.xlabel(’Hour␣Index’)
    plt.ylabel(’Sessions’)
    plt.title(’Actual␣vs.␣Predicted␣Plot’)
    plt.legend()
    plt.grid(True)
    plt.show()

    # Calculate forecast accuracy for the last 30 values
    accuracy = calculate_accuracy(test_data['Sessions'],
        train_forecast)
    print(f'ARIMA␣Model␣Error␣(MAPE)␣for␣the␣last␣30␣
        values:␣{accuracy:.2f}%')
    print('The␣lower␣the␣value␣of␣the␣Mean␣Absolute␣
        Percentage␣Error,␣the␣better␣the␣prediction␣of␣the␣
        model')
```