
# Applied Statistics 2024-25

**Final Project**

**Statistics for Business Students - ARIMA - SARIMA models**

Electrical & Computer engineering
University of Thessally, Volos

Instructor: Elias Houstis

Kakepakis Georgios - 03198
Pisxos Vasileios - 03175
Team 4

**Abstract**

This project explores the application of statistical forecasting methods to predict the number of visits to a website over a specific time period. Our dataset [1] contains six months of web traffic data for each hour of every day totaling to, about 5.000 values. We will use SARIMA and ARIMA to forecast this dataset and present the results in this report. Before showing the results we will explain how these models work and show their strengths and weaknesses. We will compare the performance of these two methods and our observations on them and on our dataset. We implemented the models in R and python programming languages using google collab for both implementations.

**Keywords:** Time Series Forecasting, ARIMA, SARIMA, web traffic

# 1 Literature Review

All the information we give for our statistical models come from these four articles [2], [3], [4], [5] that we studied before implementing them. So we explain the statistical models by summarizing the key points of these articles.

# 2 ARIMA model

## 2.1 Understanding the model

ARIMA is a form of regression analysis. It can be understood by outlining each of its components as follows.

- **Autoregression (AR)**: refers to a model that shows a changing variable that regresses to its own lagged or prior values.
- **Integrated (I)r**: represents the differencing of raw observations to allow the time series to become stationary. Stationary, meaning that the data values are replaced by the difference of these values with the previous ones
- **Moving average (MA)**: incorporates the dependency between an observation and a residual error from a moving average model applied to lagged observations.

## 2.2 ARIMA parameters

In ARIMA model we have 3 integer values: p, d and q that indicate the type of ARIMA model we are using.

- **p**: the number of lag observations in the model.
- **d**: The degree of differencing. That is, the number of times our data has been differenced.
- **q**: The size of the moving average window.

## 2.3 Arima and stationarity

Most economic and science data show trends and seasonal patterns, which means that they are not stationary. So, differencing is used to de-trend the data and make it stationary. A stationary dataset has no trends and no seasonal patterns if possible. The mean and standard deviation of that dataset are constant and the autocorrelation between values depends only on the time difference (lag) between them, not the absolute time. Non-stationary data can lead to poor model performance when we use an ARIMA model.

### 2.4 ARIMA mathematical representation

The mathematical representation of ARIMA is as follows:

$$Y'_t = c + \phi_1 Y'_{t-1} + \phi_2 Y'_{t-2} + \cdots + \phi_p Y'_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} \quad (1)$$

Where,

- $Y'_t$: The differenced and stationary time series at time $t$.
- $c$: A constant or mean of the differenced series.
- $\phi_1$, $\phi_2$ ,..., $\phi_2$: Auto regressive parameters representing the dependence on past values.
- $\epsilon_t$: The white noise error term at time t.
- $\theta_1$, $\theta_2$ ,..., $\theta_2$: Moving average parameters representing the dependence on past forecast errors.

### 2.5 Strengths and limitations of the model

Strengths

- Captures short-term dependencies well.
- Effective for uni-variate time series forecasting.

Limitations

- Requires stationary data.
- Not suitable for datasets with long term-trends or complex seasonality (we use SARIMA in those cases).
- Parameter tuning (p, d, q) can be computationally expensive for large datasets.

## 3 SARIMA model

### 3.1 Understanding the model

SARIMA stands for Seasonal Auto-regressive Integrated Moving Average. It is an extension of the ARIMA model which is a non seasonal model. It is versatile model widely used in time set forecasting designed to handle data with seasonal patterns. It captures both short-term and long-term dependencies within the dataset.

The components of the SARIMA model, (except for the S components, the others are the same as ARIMA so we just repeat them):

- **Seasonal Component**: the S represents Seasonality in the model, referring to repeating patterns in the data. This could be daily, monthly, yearly, or any other regular interval. This is the key strength of SARIMA over ARIMA.
- **Autoregression (AR)**: refers to a model that shows a changing variable that regresses to its own lagged or prior values.
- **Integrated (I)r**: represents the differencing of raw observations to allow the time series to become stationary. Stationary, meaning that the data values are replaced by the difference of these values with the previous ones
- **Moving average (MA)**: incorporates the dependency between an observation and a residual error from a moving average model applied to lagged observations.

### 3.2 SARIMA Parameters

Except for the trend parameters (p, d, q) which are the same in ARIMA and SARIMA we have 4 extra parameters (P, D, Q, m) that are called seasonal parameters. Bellow we explain those parameters.

- **P**: Seasonal auto regressive order.
- **D**: Seasonal differencing order.
- **Q**: Seasonal moving average order.
- **m**: The number of time steps for a single seasonal period.

### 3.3 SARIMA mathematical representation

The mathematical representation of SARIMA is as follows:

$$(1-\phi_1 B)(1-\Phi_1 B^s)(1-B)(1-B^s)y_t = (1+\theta_1 B)(1+\Theta_1 B^s)\varepsilon_t \qquad (2)$$

Where,

- $y_t$: The observed time series at time $t$.
- $B$: The backward shift operator, representing the lag operator.
- $\phi_1$: The non-seasonal auto regressive coefficient.
- $\Phi_1$: The seasonal auto regressive coefficient.
- $\theta_1$: The non-seasonal moving average coefficient.
- $\Theta_1$: The seasonal moving average coefficient.
- $s$: The seasonal period.
- $\varepsilon_t$: The white noise error term at time $t$.

### 3.4 Strengths and limitations of the model

Strengths

- SARIMA explicitly accounts for seasonality by incorporating seasonal auto regressive and moving average terms, making it effective for time series with repeating seasonal patterns.
- SARIMA can provide an accurate short- to medium-term forecasts once its parameters are well met.
- SARIMA's components (AR, MA, seasonal, and differencing terms) have clear statistical interpretations, which help analysts understand the underlying data dynamics.

Limitations

- Difficulty with Sudden Changes: SARIMA models struggle to adapt to sudden structural changes in the data (e.g., shocks, regime changes, or pandemics).
- The seasonal period s must be specified manually or inferred from data, which can be challenging if the seasonal frequency is not obvious
- SARIMA requires the time series to be stationary, meaning its mean and variance must be constant over time. Non-stationary data requires transformations (e.g., differencing).
- SARIMA assumes a linear relationship among past observations and errors. It may struggle with time series that exhibit non-linear or chaotic patterns.

## 4 Data Description and Prepossessing

### 4.1 Data description

Our dataset has two columns, one is the Hour Index that starts at 0 and the other is the active sessions open at the website at this particular day at this time. When we opened the dataset in python it was a dtaframe 'df' so we transformed in into a time series with hourly data starting from a specified date and time. Because we didn't know the actual month and year that the values were taken from we set the starting point as January 1, 2025 at midnight (00:00:00). Bellow you can see the plot of our dataset. In the picture it is also shown that we kept the last day of our dataset (24 hours) to forecast the model and see the accuracy of its predictions.

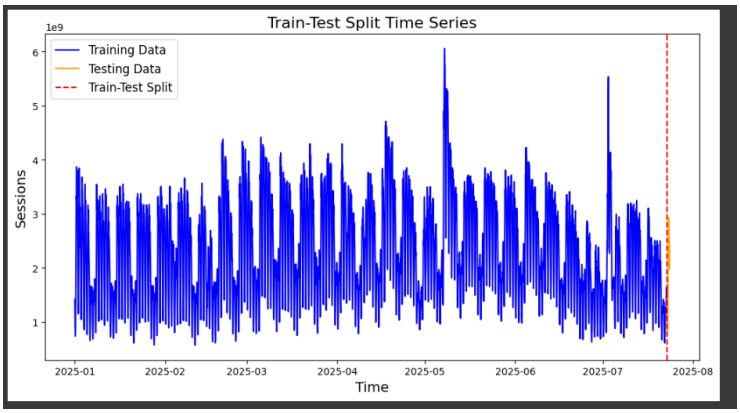


Figure 1: Our dataset and the split we did for testing and training

We plot here the testing set alone to demonstrate it better since it isn't very visible in the above plot:

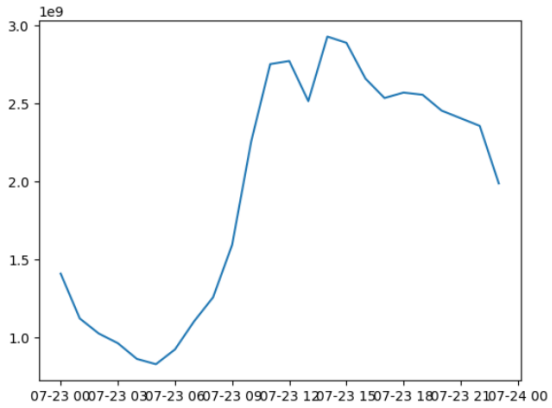


Figure 2: Test Data

## 4.2 Checking for stationarity

We used the KPSS statistical test[6] to determine if our data is stationary. Although from visually inspecting the plot we can clearly see that our data is not stationary. That is the case because because the amplitude of the fluctuations seems to vary over time with different peaks across the series, meaning that we most probably don't have a constant mean. Also there are clear seasonal patterns visible in the data and the variance doesn't appear constant throughout the time series. The KPSS test confirms our suspicions that the dataset is not in fact stationary.

### 4.3 Seasonal Decomposition and Manual Differencing of our data

Seasonal decomposition is a procedure that breaks the time series down into it's individual components: trend, seasonality and residuals. We did seasonal decomposition on our dataset and got the results shown in the plot bellow:

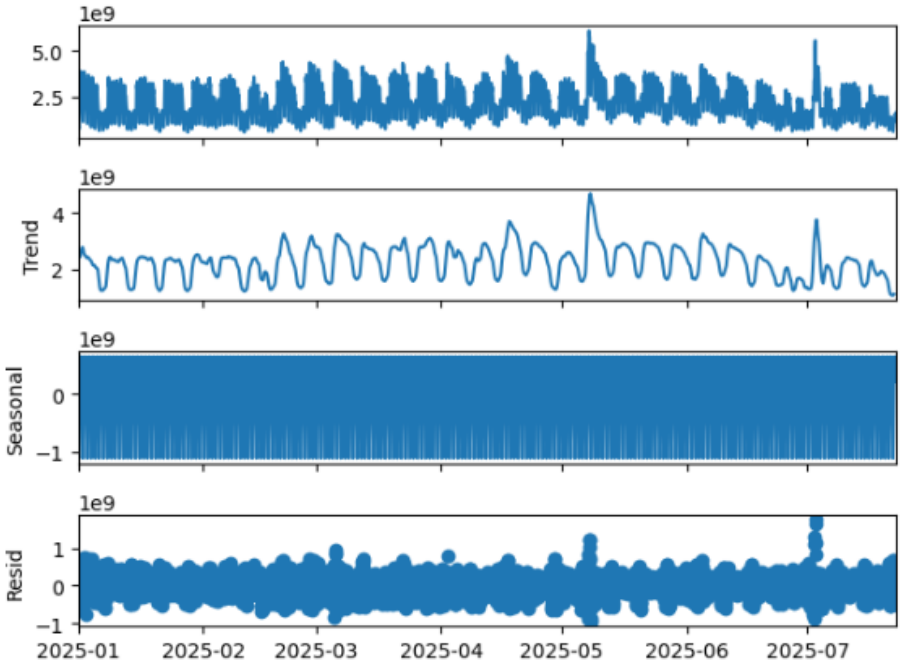


Figure 3: The seasonality, trend and residuals of our time series

We also plotted the diagrams for autocorrelation and partial autocorrelation for the Sessions in our time series. Autocorrelation measures the linear relationship between a time series and it's lagged values. It shows how much the value at time t is correlated with its value at time t - k where k is the lag. Partial autocorrelation measures the correlation between the time series and its lagged values, excluding the influence of intermediate lags. That means that the correlation at lag k accounts for the direct effect of the lag k, removing the effects of lags 1,2,...,k-1. The shaded region in the auto correlation plot represents the confidence intervals. The correlation values that exceed this shaded region are statistically significant. Below we present these plots.

The plots also indicate non-stationarity in our data. The autocorrelation plot suggests that the data has periodic or seasonal components which could be caused by repeating cycles in the time series. The gradually decreasing correlation also indicates that the mean and variance of the series is likely changing over time. A stationary series typically has an auto correlation plot that cuts

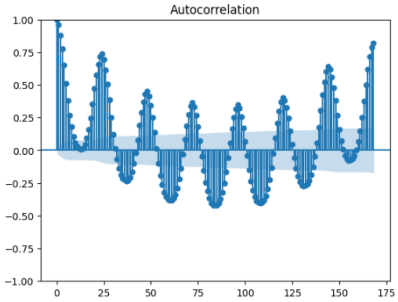


Figure 4: Autocorrelation

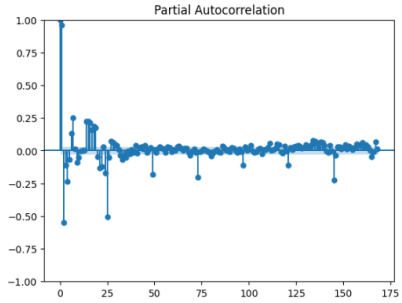


Figure 5: Partial Autocorrelation

off quickly. From the partial auto correlation plot we can deduct that that only the first few lags are significant. This suggests that the immediate past values influence the current value. From both plots we get crucial information about the ARIMA's and SARIMA's parameters that we are going to analyze further later in the report. We continue our dataset analysis by performing differencing[7] in our dataset. This is a method used to transform a non-stationary into a stationary series by subtracting the current value from the previous value. The process of differencing eliminates changes in the level of a time series and reduces trends, making it easier to model and predict the time series. We performed first level differencing (meaning that we subtracted the current value of the time series from the previous value) and then we used the KPSS[6] test to see if the series was stationary, or if we needed to do second differencing(subtracting the current value of the time series from the two previous value). Below is a plot of our time series after the differencing.

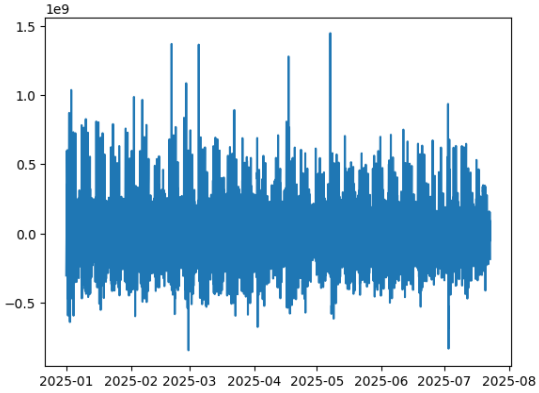


Figure 6: Our time series after differencing

We also did seasonal decomposition after the differencing and plotted the

autocorrelation and partial autocorrelation of our series. Below we see these plots.

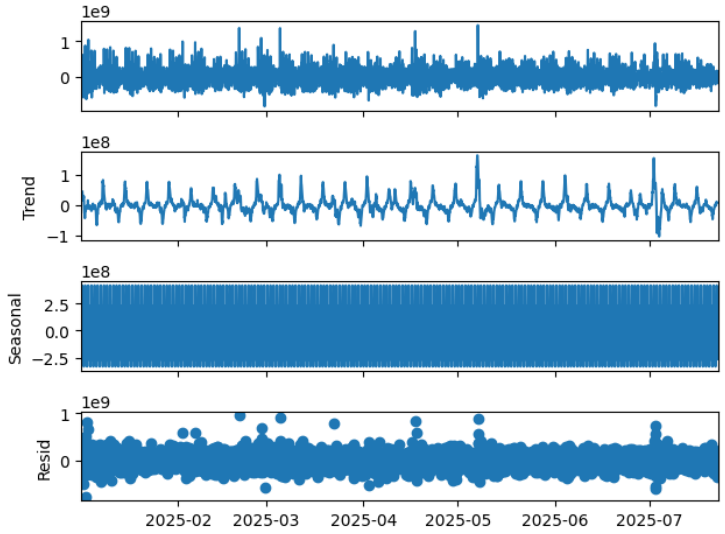


Figure 7: The seasonality, trend and residuals of our differenced time series

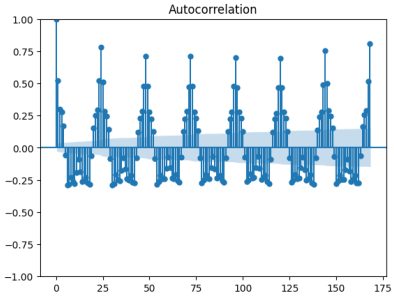


Figure 8: Autocorrelation after differencing

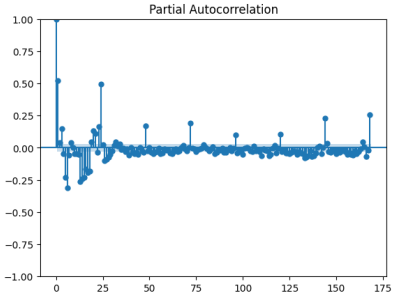


Figure 9: Partial Autocorrelation after differencing

From visual inspection we can understand that the trend in the original data appears to have been eliminated by differencing because there is no gradual decay in the auto correlation function now. But the periodic patterns still remain, meaning that the data definitely still has seasonal components. So to have a good forecasting model we might need to apply seasonal differencing (something that the SARIMA model applies). From the partial auto correlation plot we can deduct that that only the first few lags are significant, something that we also saw on the plots before differencing. After inspecting the plots, we

do the KPSS test one more time and it confirms that our time series is now in fact stationary, it just still has seasonality.

## 4.4 Detecting the seasonal period of our time series

To detect the seasonal period of our dataset we use Fast Fournier Transform(FFT)[8]. This method detects the dominant frequency in our time series and derives it's seasonal period. It converts the time-domain representation of our data into a frequency-domain representation and the dominant frequency corresponds to the strongest repeating pattern in the data. By using this method we detect that the seasonal period in our series is 24 hours. That means that our time series has daily seasonality.

## 4.5 Performing Seasonal Differencing Manually

After detecting the dominant frequency of our time series we performed seasonal differencing. Seasonal differencing is the same as regular differencing but it represents the difference between a value and a value with lag that is multiple of S. S is 24 in our case. Here is the plot of our seasonally differenced time series against the starting one.


Figure 10: The starting time series and the time series after seasonal differencing

We also plotted the auto correlation and partial auto correlation of the seasonally differenced dataset.

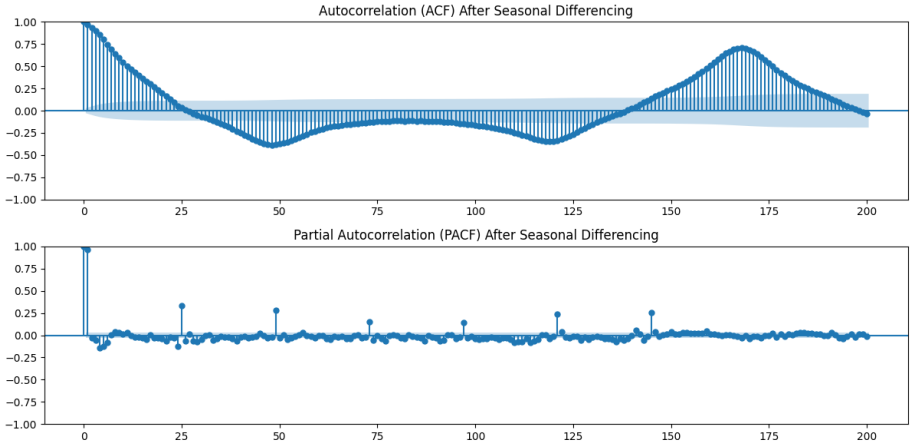


Figure 11: Auto correlation and partial auto correlation of seasonally differenced dataset

# 5 Implementing ARIMA and SARIMA models

## 5.1 Implementing ARIMA model

The ARIMA model has three parameters(p,d,q) that we have already explained earlier in our report. By examining the partial auto correlation plot after differencing we see that there is a sharp cut-off on the lag 2. So we select p = 2. We examine the autocorrelation function as well and see that there is a sharp cut off at lag 5 so we pick q = 5. Lastly we know that after a single differencing the data is stationary so we pick d = 1.

## 5.2 Implementing SARIMA model

The SARIMA model has four more parameters than the ARIMA model (P, D, Q, s) that we have already explained earlier in our report. By examining the partial auto correlation plot after seasonal differencing we see that there is a sharp cut-off on the lag 2. So we select P = 2. We examine the autocorrelation function as well and see that there is not a visible cut off at any lag so we pick Q = 1 because we assume that the biggest drop is there. We know from the Fast Fourier Transform (FFT) that s = 24 and lastly we know that after a single seasonal differencing the data is stationary so we pick D = 1.

# 6 Results - Comparison

We implemented both models in R and Python. In both cases **SARIMA** outperformed ARIMA, which was expected since the data is not stationary. In this section we compare the 2 models, using both the Python and the R software.

## 6.1 ARIMA vs SARIMA - Python

First we plot the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) values of the two models. AIC and BIC are metrics that reflect how well a model fits the **training data** and also incorporate a penalty for model complexity to avoid overfitting. Since theys use a very large scale the difference is not visible by comparing the plot of the actual values, so we also include logarithmic-scale plot:

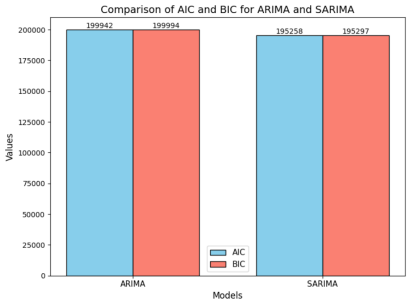


Figure 12: AIC/BIC (real scale)

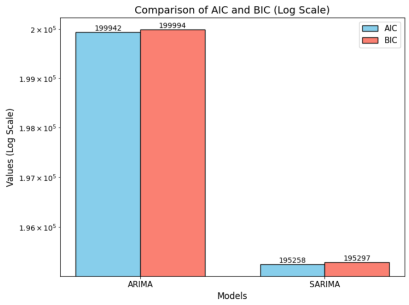


Figure 13: AIC/BIC (log-scale)

As we see in both cases the SARIMA model has a lower AIC and BIC value, suggesting better performance of the model on the training data with an appropriate level of complexity. Next, we want to compare the **forecasting values** of the models with the actual test valeus. We first visualize the forecast of each model:

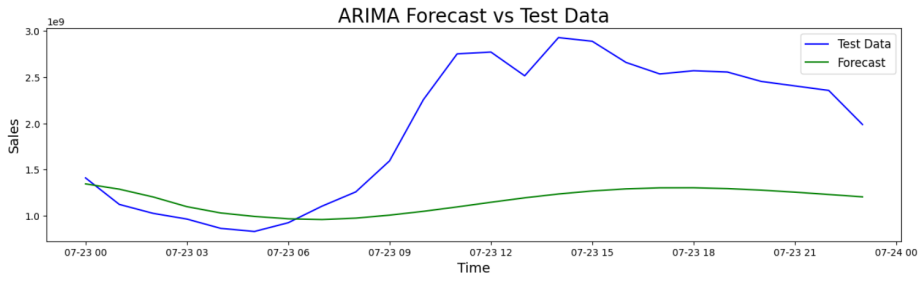


Figure 14: ARIMA: forecasts VS test values

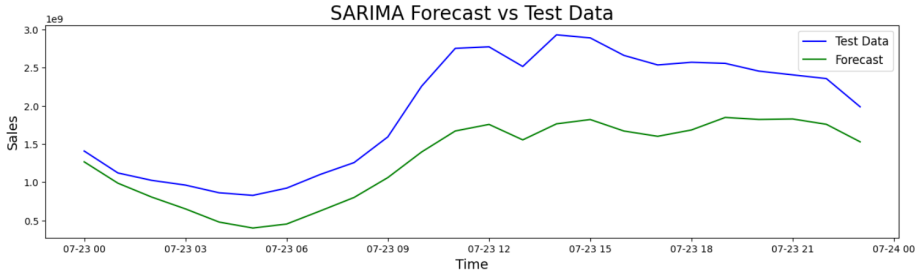


Figure 15: SARIMA: forecasts VS test values

We can already see by looking at the visualization that the SARIMA model makes better predictions. Finally, we use the following metrics to compare the performance of the two models: MAE, MSE, RMSE, MAPE.

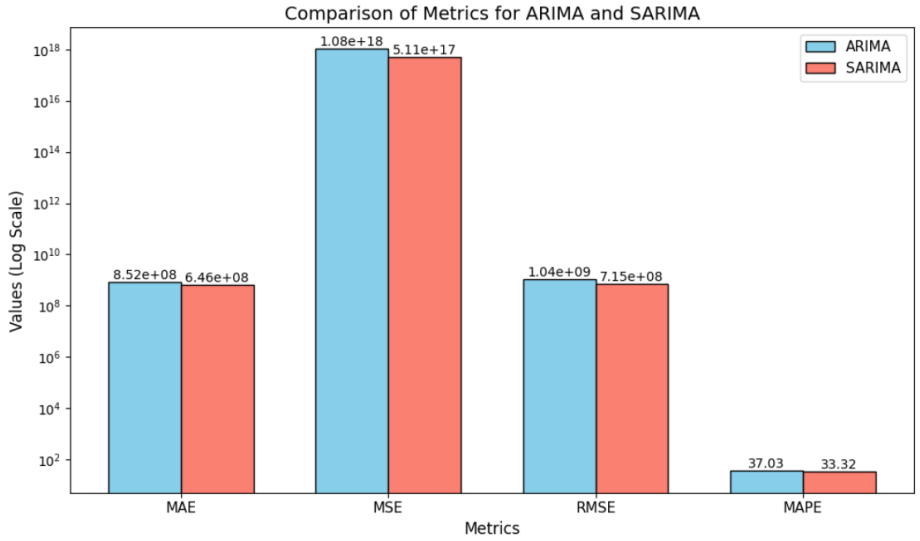


Figure 16: ARIMA vs SARIMA metrics

We see that in every metric the SARIMA model has a smaller error value. Through all the metrics and the visualizations we obtain, we conclude that the SARIMA model performs better than the ARIMA model.

## 6.2 ARIMA vs SARIMA - R

We perform the same procedure in R now. First, let's plot again the AIC and BIC value we get using R instead of python. Again we plot both the original and the logarithmic scale to demonstrate the difference:

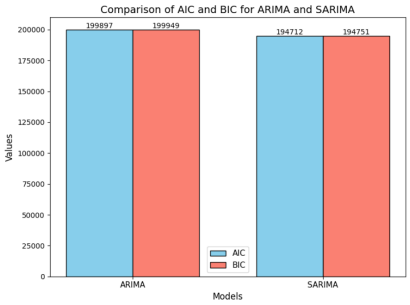


Figure 17: AIC/BIC (real scale)

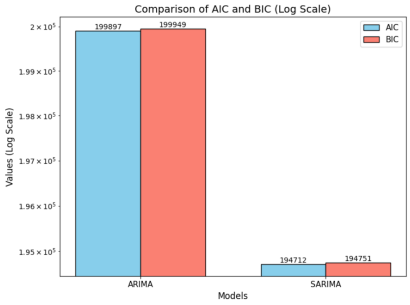


Figure 18: AIC/BIC (log-scale)

We see very similar results with python. The SARIMA model is better, and the values are close to the values extracted by python. We next compare the forecasts:

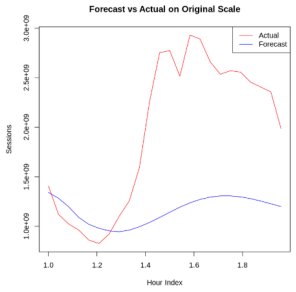


Figure 19: ARIMA: forecasts VS test values

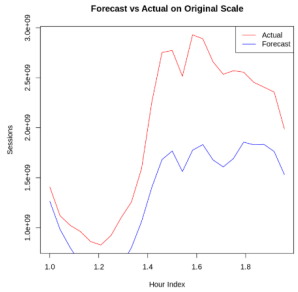


Figure 20: SARIMA: forecasts VS test values

Again we see the same behavior with python, although the plots are not a little narrower so the scale looks a little different. So check for certain we use the following metrics in R to compare the two models: MAE, RMSE, MAPE,

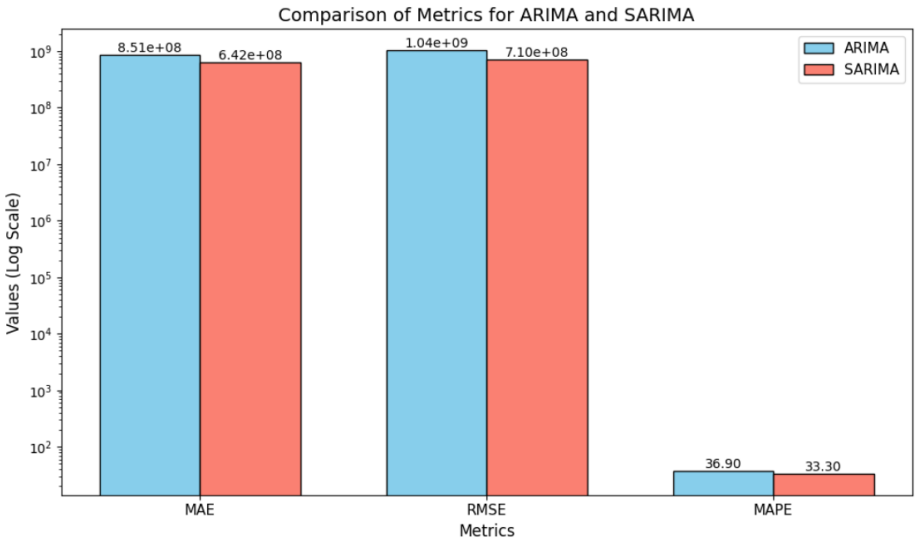


Figure 21: ARIMA vs SARIMA metrics

In this case also we see that SARIMA outperforms ARIMA, and again the numbers are very similar with the corresponding error values we obtained in python.

# 7 Conclusions

In this project we uses a web-trafficking dataset which represents the number of visits to a website over a specific period of time. We applied time-series forecasting on the data using the ARIMA and the SARIMA models. Initially, we inspected the dataset and understood that it is **non-stationary**. We also ploted the ACF/PACF plot for ordinal and seasonal differcing in order to find the optimal parameters for the models we used, and we ended up with: **ARIMA(5, 1, 2)** and **SARIMA(1, 1, 1)(2, 1, 1)(24)**.

Using both Python and R softwares and multiple criteria (AIC/BIC/RMSE/ MAE/MAPE/...) we conclude that the **SARIMA** model outperforms the ARIMA model. This makes sense since the data was non-stationary and ARIMA models struggle to capture the patterns on these datasets.

# References


[1] "Our web traffic dataset", available at: https://www.kaggle.com/datasets/kajal1/web-traffic-forecast-dataset.

[2] "info on SARIMA", available at: https://www.geeksforgeeks.org/sarima-seasonal-autoregressive-integrated-moving-average/.

[3] "info on SARIMA", available at: https://machinelearningmastery.com/sarima-for-time-series-forecasting-in-python/.

[4] "info on ARIMA", available at: https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp.

[5] "info on ARIMA", available at: https://www.geeksforgeeks.org/model-selection-for-arima/.

[6] "Info about the kpss test", available at: https://www.statisticshowto.com/kpss-test/,

[7] "Information about stationarity and differencing", available at: https://otexts.com/fpp2/stationarity.html,

[8] "Information about Fast Fournier Transform", available at: https://www.nti-audio.com/en/support/know-how/fast-fourier-transform-fft,