**NIVEDITA**

**Data Pre-processing**

Data pre-processing is a fundamental step in data analysis and machine learning that involves transforming raw data into a format that is clean, consistent, and suitable for analysis. This step is critical because the quality of the data directly impacts the effectiveness of the analysis and the accuracy of predictive models. In this report, we focus on the pre-processing steps applied to attendance data collected over several years across different datasets.

**Summary of the raw data:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Dataset** | **Columns** | **Rows** | **Important Information** |
| D19  (Target Audience- IT Managers) | **5** | 1185 | 1. Details of the year of 2019  2. Details of ‘Attended’ column:  Yes- 839  No- 247  NaN- 99  3. Details of ‘Attendee Status’:  Attending- 1082  Cancelled- 90  Booker not attending- 13 |
| D21  (Target Audience- IT Managers) | **5** | 671 | Details of the year of 2020 and 2021  2. Details of ‘Attended’ column:  Yes- 340  NaN- 331  3. Details of ‘Attendee Status’:  Attending- 669  Cancelled- 2 |
| GP21  (Target Audience- Property Managers | **5** | 798 | Details of the year of 2020 and 2021  2. Details of ‘Attended’ column:  NaN- 798  3. Details of ‘Attendee Status’:  Attending- 767  Cancelled- 31 |
| MSE21  (Target Audience-  Education Property Managers | **5** | 1601 | Details of the year of 2021  2. Details of ‘Attended’ column:  NaN- 1601  3. Details of ‘Attendee Status’:  Attending- 1582  Cancelled- 19 |
| NP21  (Target Audience- Property Managers | **5** | 401 | Details of the year of 2020 and 2021  2. Details of ‘Attended’ column:  Yes- 280  NaN- 93  No- 28  3. Details of ‘Attendee Status’:  Attending- 401 |
| SRM22  (Target Audience- Education Managers | **5** | 998 | Details of the year of 2022  2. Details of ‘Attended’ column:  NaN- 722  Yes- 196  No- 80  3. Details of ‘Attendee Status’:  Attending- 998 |
| SRM23  (Target Audience- Education Managers | **5** | 753 | Details of the year of 2023  2. Details of ‘Attended’ column:  NaN- 405  Yes- 300  No- 48  3. Details of ‘Attendee Status’:  Attending- 741  Cancelled- 12 |

**Overview of Data Pre-Processing Steps**

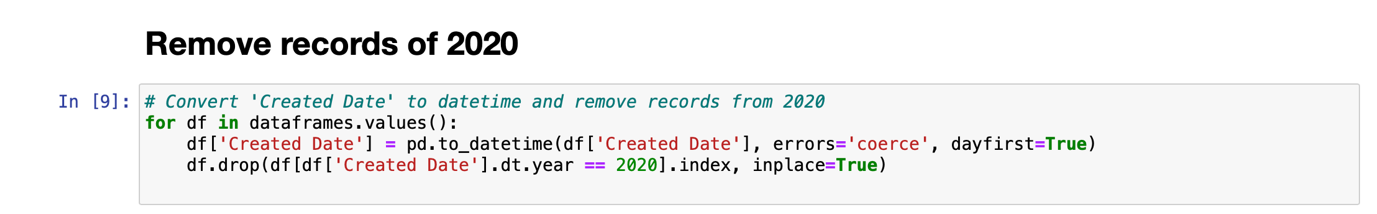
Data pre-processing is essential for optimizing the accuracy and efficiency of data analysis and machine learning models. In this project, we implemented several key pre-processing steps:

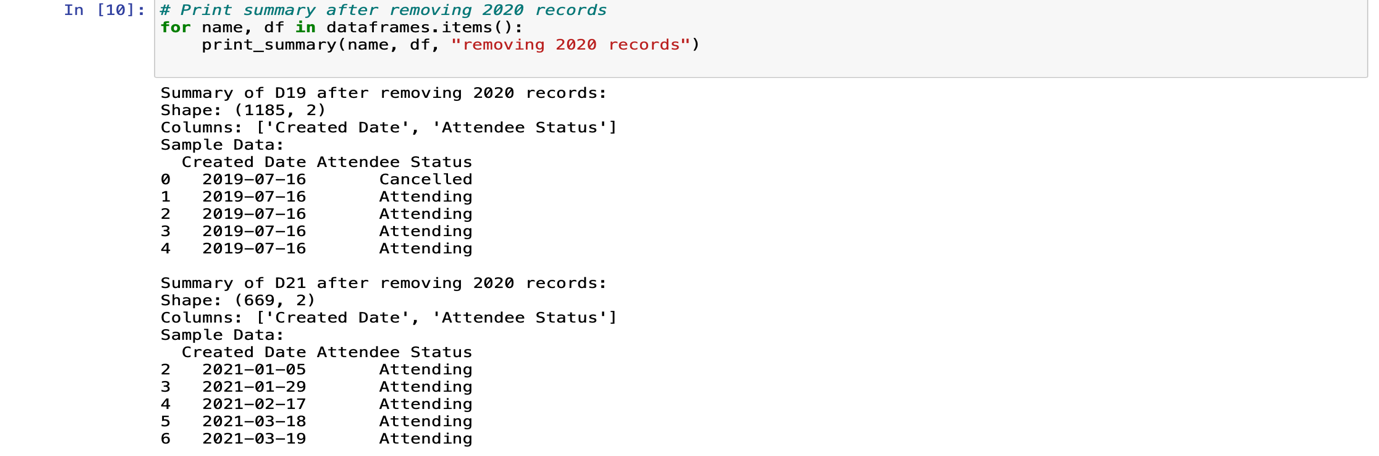
1. **Removing Unnecessary Columns:**

* **Purpose:** Simplifies the dataset by eliminating irrelevant features that do not contribute to the analysis or predictive modeling. This reduces the complexity and computational load during processing.
* **Impact:** Streamlining the data helps focus the model on relevant features, which can improve performance and reduce the risk of overfitting.

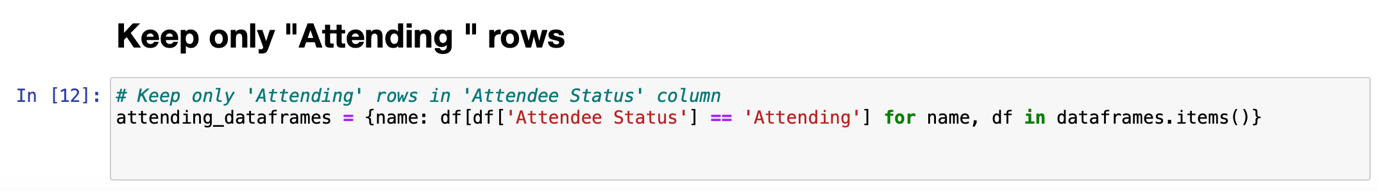
After the second week meeting with client, we decided to remove-

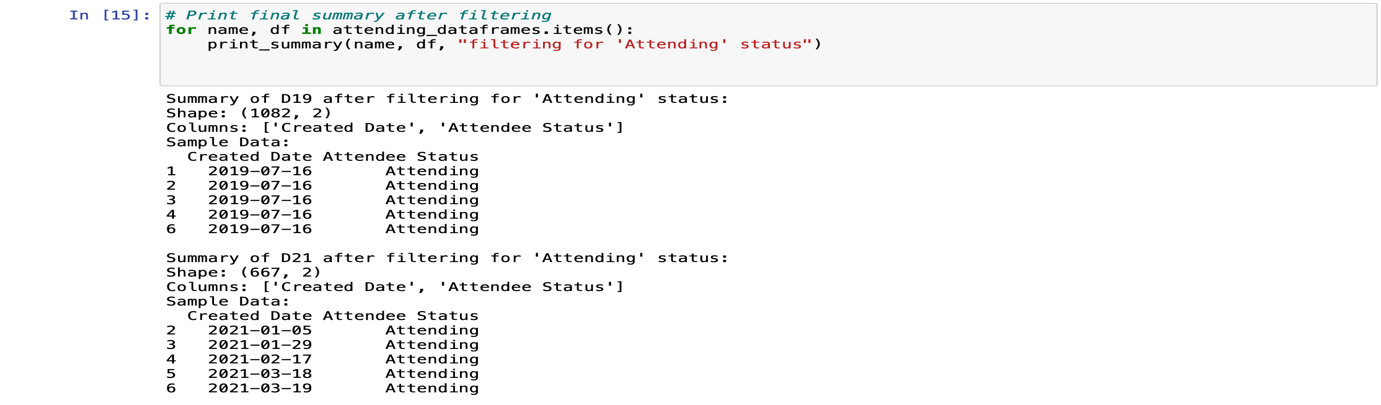
* *2020 records:* This was the period when pandemic was going on, which might be affected the predictive model indirectly.





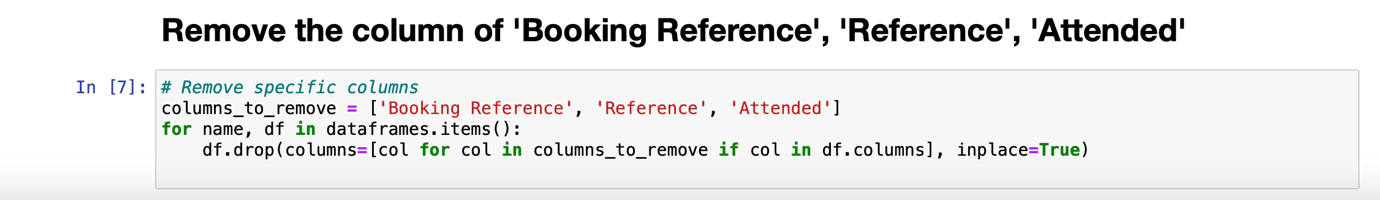
* *‘Attended’ column:* In this column almost 4049 entries out of 6407 falls into the category of ‘NaN’, which stands for ‘Not a Number’ and is typically used in datasets to denote missing or undefined values.

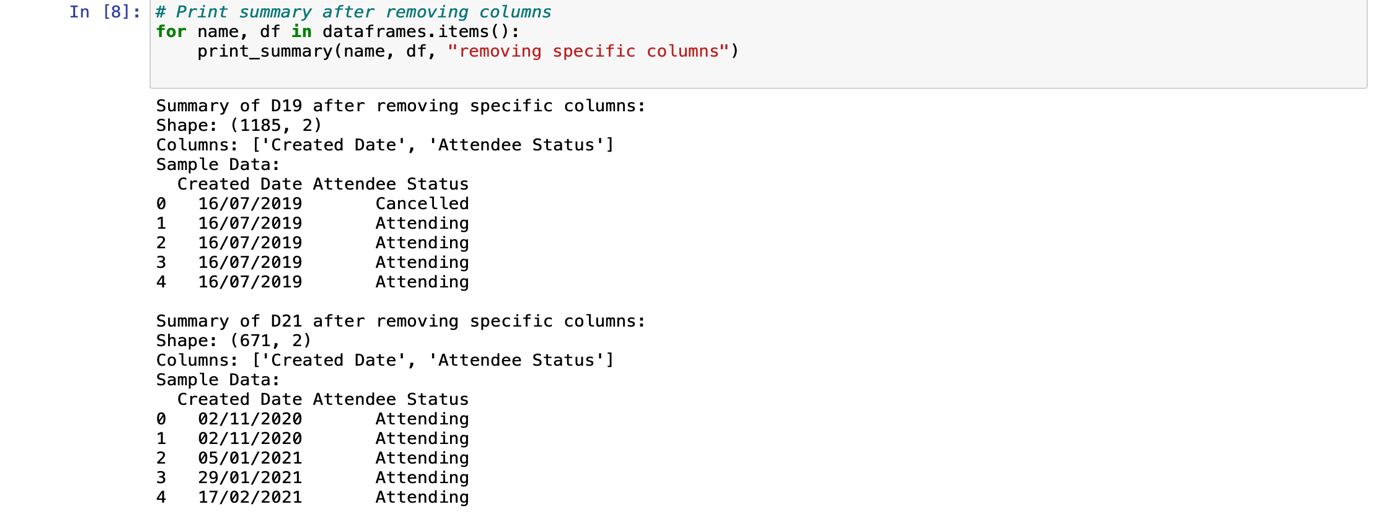




The presence of total 403 ‘No’ entries could be of interest depending on the context of the event. For example, if attendance is crucial, these ‘No’ entries could warrant further investigation or follow-up. Based on the given details, approximately 63% data related to ‘Attended’ column are missing, 6% of the total entries are fall into section of ‘No’. So, we decided to delete the column instead of doing the imputation. There are several ways to deal with missing values, like- Mean/Median/Mode Imputation, K-Nearest Neighbours (KNN), Interpolation. This method is generally not recommended when the missing data proportion is this high, as it artificially inflates the sample size with potentially misleading values. When almost 63% of data are missing from a dataset, handling this significant amount of missing data becomes especially challenging, and the methods used can have a substantial impact on the reliability and validity of the analysis.

* *‘Attendee Status’ column:* In this column we only focused on the ‘Attending’ status. After discussing with client, we realized that the main goal of this project is how many people are going to attend the conference instead of cancel the booking. To make a robust prediction model we decided to delete the entries other than ‘Attending’.
* *‘Booking Reference’ and ‘Reference’ column:* In this project, these two columns have no significance with the prediction. Due to this reason, we cleaned the entries related to these columns.





**2. Data Aggregation:**

We aggregated the data by dates and counting occurrences, which is a common method in data analysis to simplify and summarize data, making it easier to observe trends and patterns over time. This transformation is particularly useful for time series forecasting or any analysis where the frequency of events within specific time frames is relevant. It helps in understanding the distribution and trends of data points across different dates.

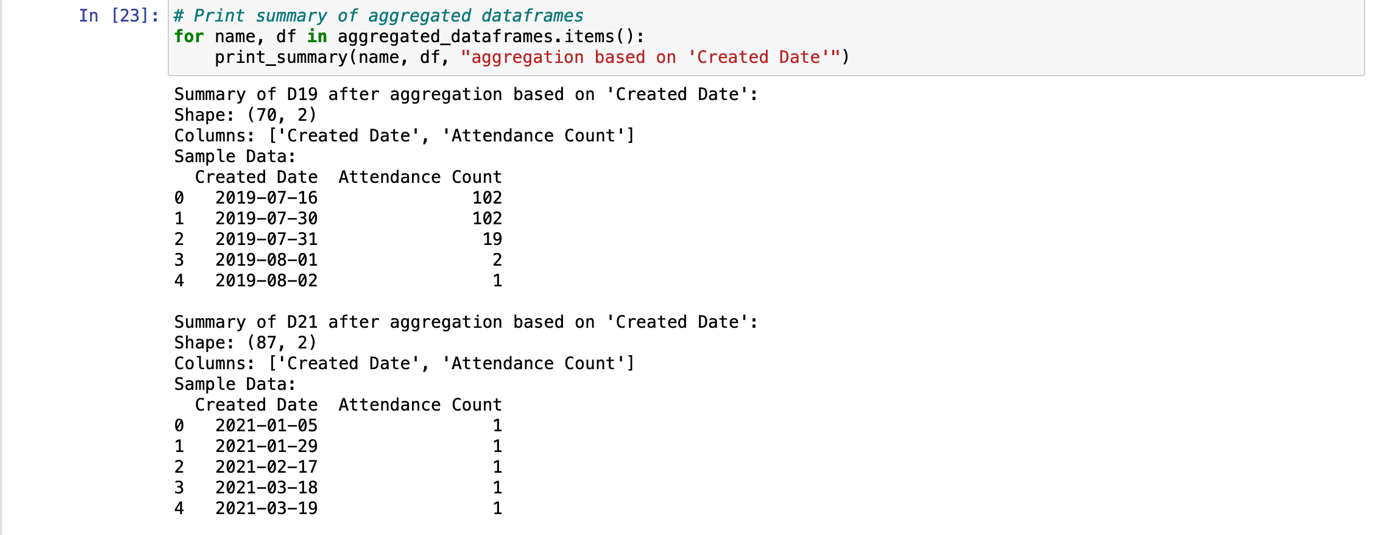
### Purpose:

1. ***Simplifying Data****:* Aggregation reduces the complexity of data by summarizing detailed data into a more manageable form. This is especially useful in handling large datasets where individual data points might be less informative than their summaries.
2. ***Enhancing Performance****:* By reducing the volume of data through aggregation, computational efficiency can be significantly improved. This is critical in scenarios involving large-scale data processing where performance and speed are concerns.
3. ***Improving Data Quality****:* Aggregation can help smooth out noise in the data by averaging out misleading fluctuations. This leads to cleaner and more reliable datasets.
4. ***Revealing Patterns****:* Aggregating data over time, geographical regions, or other dimensions can reveal trends and patterns that are not observable in raw, granular data. For example, aggregating sales data by month can show seasonal trends that daily sales data might obscure.

* **Impact:** Aggregation can either enhance or diminish the accuracy of insights drawn from the data. While it can clarify trends, over-aggregation can lead to loss of critical details. The key is to find the right level of aggregation that balances detail with clarity.

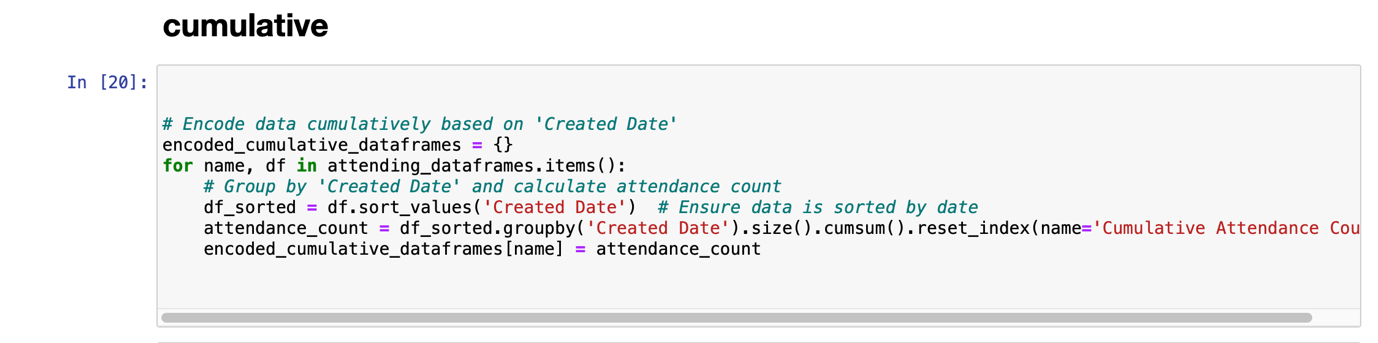


The function aims to aggregate attendance data by counting the number of occurrences for each 'Created Date' in multiple dataframes and store the aggregated results in new dataframes.



# *Cumulative:*

The function aims to aggregate attendance data by calculating the cumulative count of occurrences for each 'Created Date' in multiple dataframes and store the cumulative results in new dataframes.



By applying these pre-processing steps, we ensure that the data is clean, structured, and ready for further analysis or machine learning modelling. These steps help in extracting meaningful insights and building robust predictive models.

In the realm of time series forecasting, the AutoRegressive Integrated Moving Average (ARIMA) model stands out due to its ability to capture the intricate patterns in sequential data. This report delves into the application of ARIMA modeling for predicting conference attendance, leveraging historical attendance data to inform future trends. ARIMA's flexibility allows it to handle non-stationarity through differencing, while its autoregressive (AR) and moving average (MA) components model the dependencies between observations. Given the notable fluctuations and potential seasonal effects in the dataset, ARIMA, and its seasonal variant SARIMA, are explored to provide accurate forecasts essential for planning and resource allocation.

**WHEN WE USE ARIMA?**

ARIMA, which stands for AutoRegressive Integrated Moving Average, is a widely used statistical method for time series analysis and forecasting due to its ability to handle non-stationarity by incorporating differencing, capture autocorrelation through its autoregressive (AR) and moving average (MA) components, and model seasonal effects with its extension to Seasonal ARIMA (SARIMA). Its flexibility allows it to be applied across various types of time series data, such as financial markets, sales data, and economic indicators. ARIMA’s solid statistical foundation provides clear guidelines for model identification, estimation, and diagnostic checking, making it a reliable and interpretable method. The model is well-documented, widely understood, and supported by numerous software tools, which contributes to its popularity. Additionally, ARIMA models often deliver accurate forecasts and can be combined with other methods, such as machine learning techniques, to enhance forecasting performance, making it a versatile and powerful tool for statisticians, economists, and data scientists.

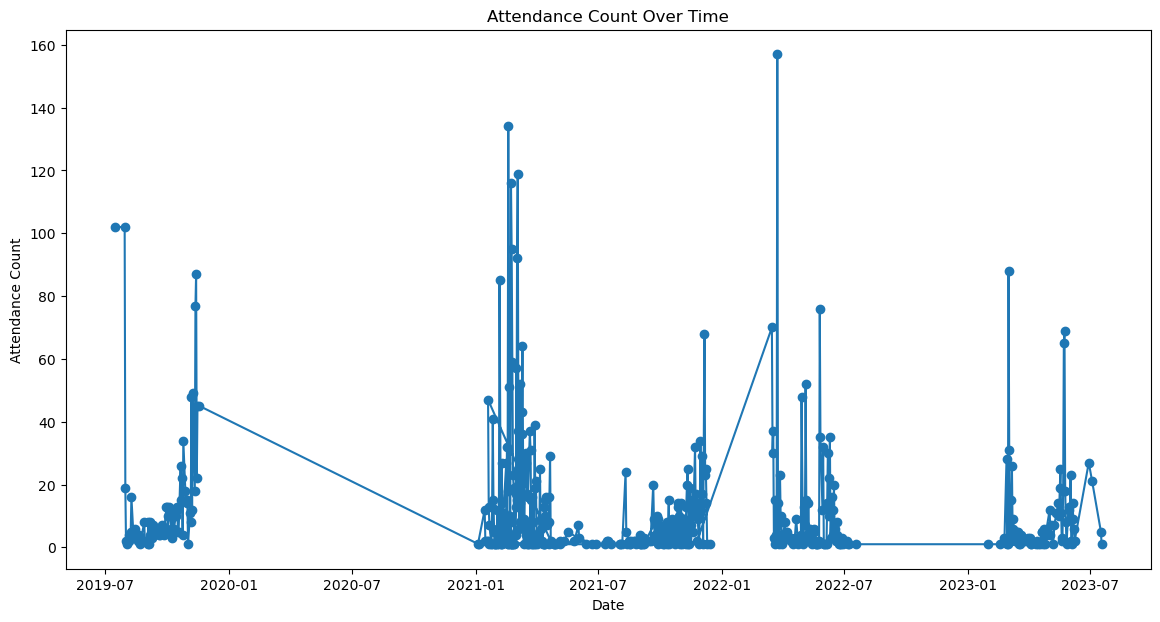
**Rational to use ARIMA related to this project-** Using ARIMA to predict conference attendance is highly relevant to this project goal, given its strengths in time series forecasting. Because ARIMA's ability to capture patterns in historical attendance data can provide accurate forecasts, essential for planning and resource allocation.

*Visualize the data:*

The goal of this project is to predict conference attendance, and visualizing the data is a crucial step in achieving this goal.



Visualizing the attendance count over time is crucial for understanding trends and patterns, as it allows us to easily identify seasonal effects, anomalies, and overall trends in the data. This understanding is essential for selecting appropriate forecasting models. The plot helps determine whether attendance is increasing, decreasing, or remaining constant, informing future strategies and decisions. Additionally, visualizing the data enables a preliminary assessment of stationarity, which is necessary for ARIMA modeling, and helps identify recurring patterns or seasonal effects that might guide the inclusion of seasonal components in the model or suggest the need for alternative models.

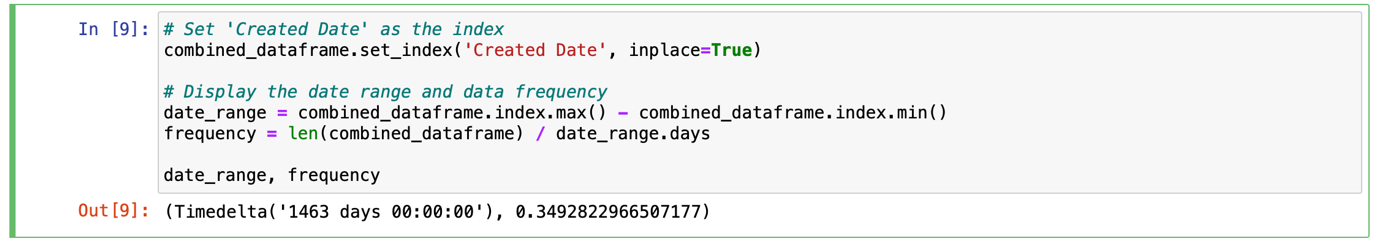


*Explanation:* The graph shows significant fluctuations in attendance over the years, with some periods of high attendance followed by periods of very low attendance.There are noticeable gaps particularly around the end of 2020 and beginning of 2021. This is the indicative of external factors affecting attendance (e.g., the impact of COVID-19).

Several spikes exceed 100 in attendance, particularly in 2021 and 2022. These could represent special events or anomalies where attendance was unusually high. The drop to nearly zero attendance around the end of 2020 and early 2021 is significant and may warrant further investigation.

*'Created Date' as index:*

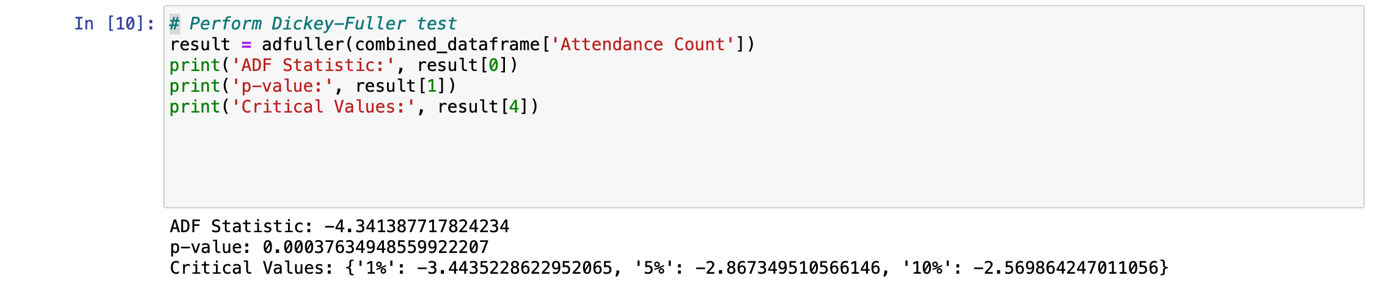
Setting the 'Created Date' as the index is a standard preprocessing step for time series data, enabling time-based operations and analysis. Calculating the date range ensures that the data covers a sufficient period for meaningful analysis and forecasting. Determining the data frequency provides insights into how often the data is recorded, which is essential for configuring the time series model (e.g., ARIMA) and understanding whether the data points are recorded daily, weekly, or at irregular intervals. Knowing the date range and data frequency assists in selecting the right parameters for the ARIMA model and in determining if any preprocessing, such as resampling or interpolation, is needed.



It indicates that the dataset spans 1463 days (about 4 years) and has an average frequency of approximately 0.35 data points per day, meaning there is roughly one data point every three days.

*Dickey-Fuller test, check the p-value:*

The Dickey-Fuller test is used in the context of ARIMA (AutoRegressive Integrated Moving Average) modeling to check for stationarity in a time series. ARIMA models require the time series data to be stationary. Stationarity means that the statistical properties of the series (mean, variance, autocorrelation, etc.) are constant over time. Non-stationary data can lead to unreliable and inaccurate model predictions. Non-stationarity can manifest as trends, seasonality, or other time-dependent structures in the data.

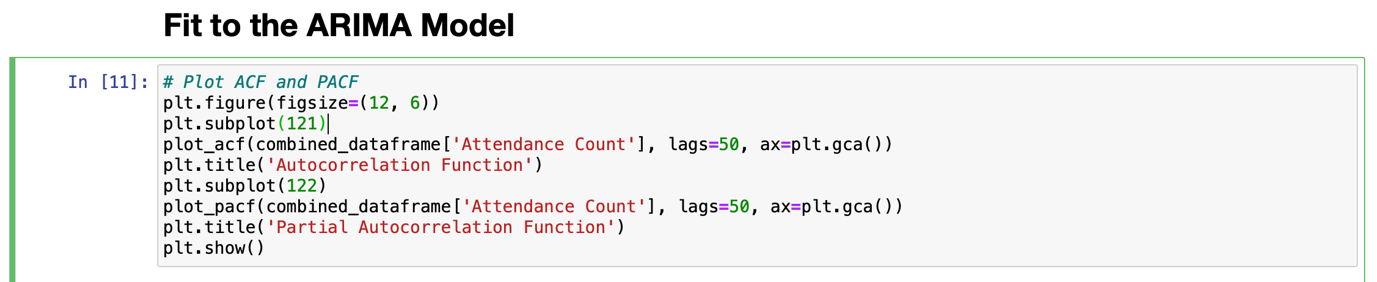


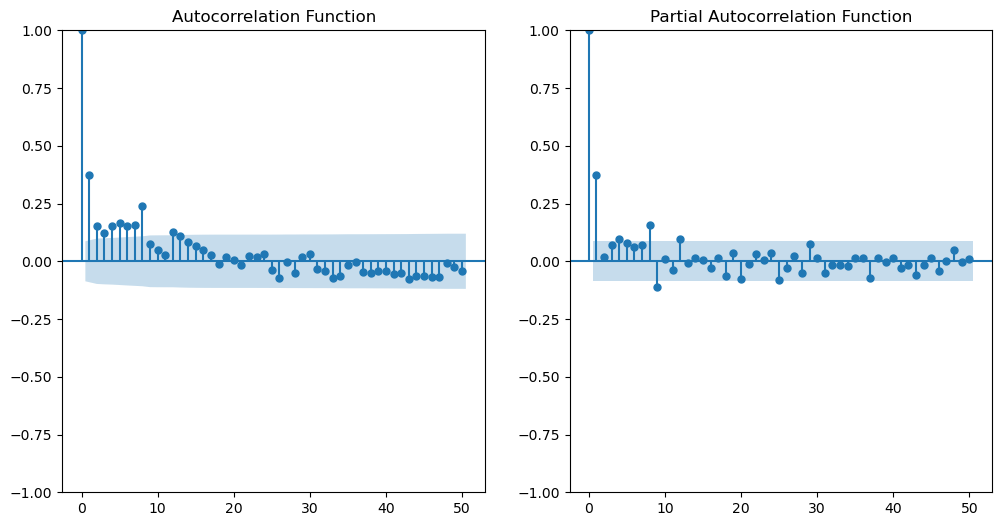
Since the ADF Statistic is less than all the critical values at the 1%, 5%, and 10% significance levels, and the p-value is less than 0.05, we reject the null hypothesis. This means:

The time series is stationary. No differencing is needed for the ARIMA model. Given that the time series is already stationary, we can proceed with fitting an ARIMA model without applying differencing.

*ACF and PACF:*

The ACF and PACF plots provide essential insights into data patterns by revealing underlying trends and seasonality, which are crucial for configuring the ARIMA model correctly. They can also be used after fitting the ARIMA model to check the residuals' ACF and PACF, helping diagnose if the model has captured the data's structure adequately. By offering a visual representation of autocorrelations, these plots aid in making informed decisions about the model's parameters, ultimately leading to more accurate and reliable forecasts.





ACF (Autocorrelation Function):

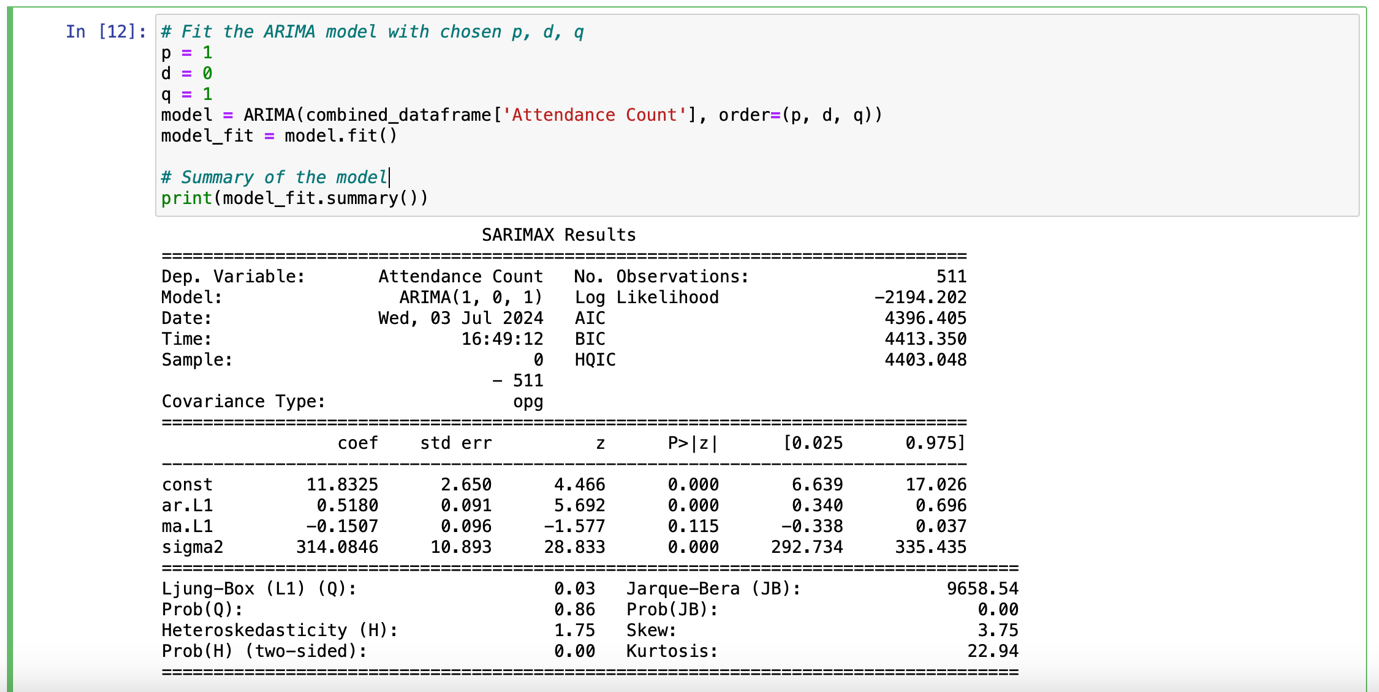
The ACF plot shows the correlation of the time series with its own lagged values. If there is a sharp drop after lag 1, it indicates that the series is stationary and has a short memory of past values. From the ACF plot, it looks like the ACF value drops significantly after lag 1 and gradually decreases, which suggests a potential MA (Moving Average) component.

PACF (Partial Autocorrelation Function):

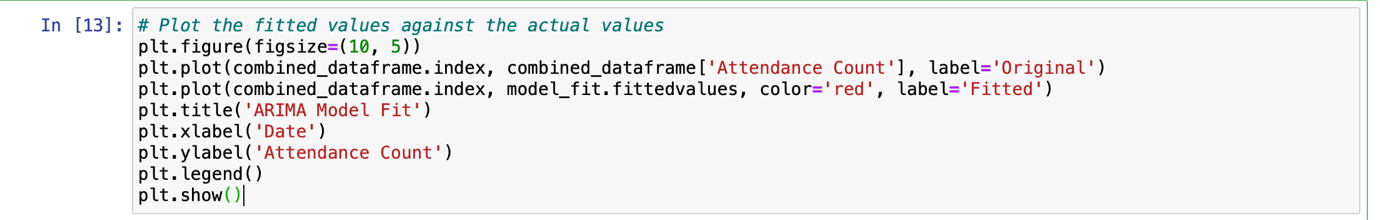
The PACF plot shows the partial correlation of the time series with its own lagged values, after removing the effects of intermediate lags. A sharp drop after lag 1 in the PACF plot suggests that the AR (AutoRegressive) process is of order 1. From the PACF plot, the partial autocorrelation drops after lag 1, suggesting an AR component.

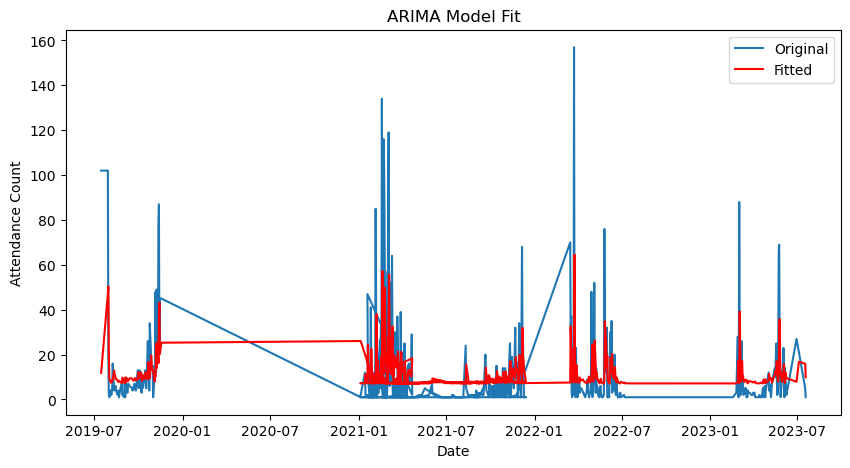
*Output of the ACF and PACF:*

p=1 and q=1 for the ARIMA model. Since we have already determined the series is stationary, d=0.



The output indicates that the ARIMA(1, 0, 1) model fits the 'Attendance Count' data, with the autoregressive term being significant, while the moving average term is not. The diagnostics suggest no significant autocorrelation in the residuals but highlight issues with normality and heteroskedasticity, which might affect the model's predictive performance. Further investigation and possibly model adjustments are needed to address these issues for improved forecasting accuracy.



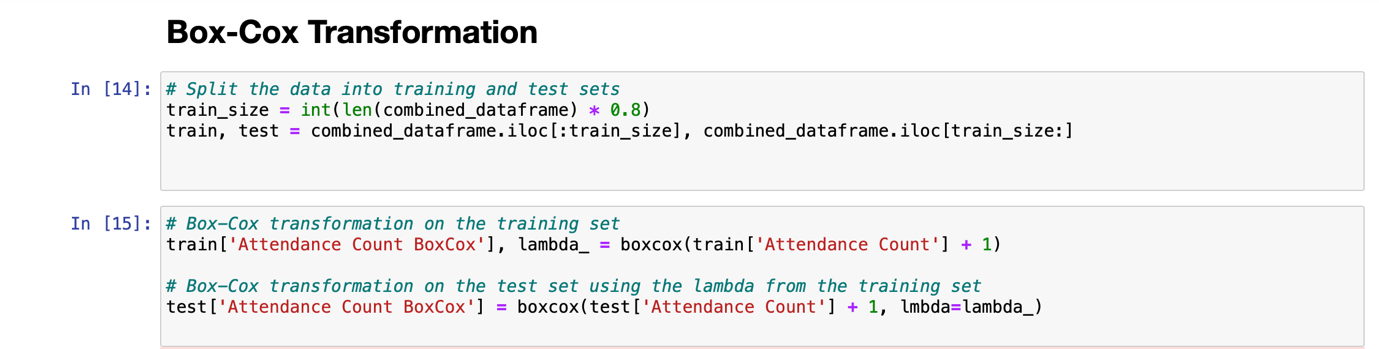


X-Axis (Date) The x-axis represents the timeline from July 2019 to July 2023. It shows the progression of time over the span of four years. Y-Axis (Attendance Count) The y-axis represents the attendance count, which ranges from 0 to 160. It measures the number of attendees recorded on each date. Data Lines Blue Line (Original): This line represents the original attendance count data. It shows the actual recorded attendance over time. Red Line (Fitted): This line represents the values predicted by the ARIMA(1,0,1) model. It shows the model's fit to the actual data.

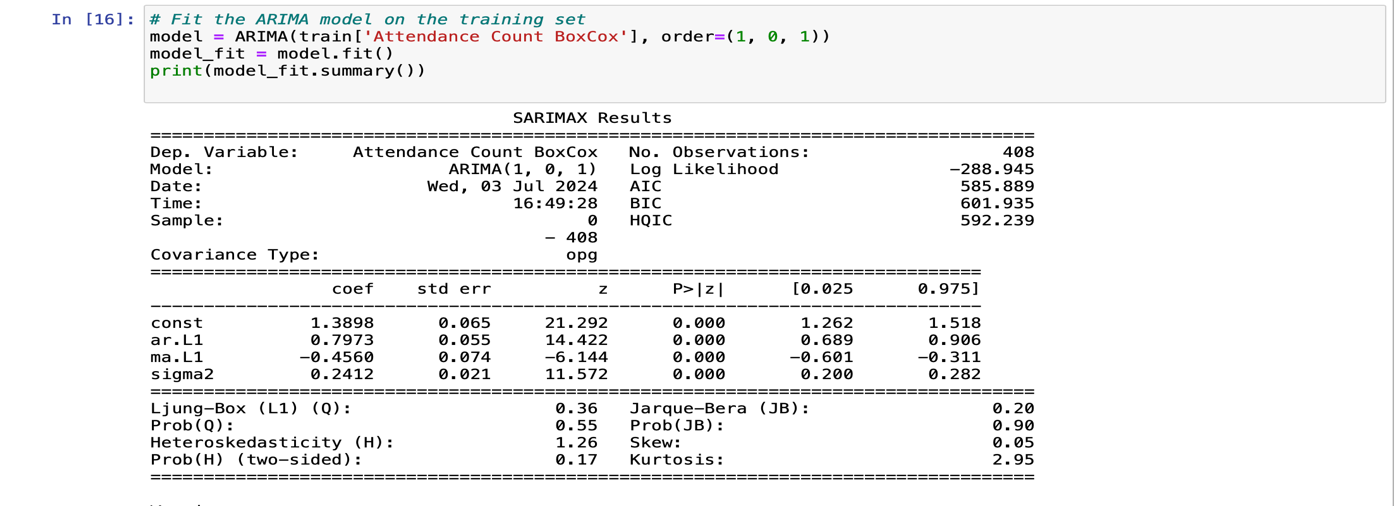
The ARIMA(1,0,1) model provides a smoothed representation of the attendance count data. The model captures the general trend and direction of the data but struggles with sharp peaks and sudden changes, which is typical for simple ARIMA models. This smoothing effect suggests that while the model is useful for understanding the general pattern, it might not be the best choice for data with high variability and frequent extreme values.

*Box-Cox transformation:*

The plot indicates that the ARIMA model's fitted values (red line) do not closely match the actual attendance counts (blue line) in several places, particularly where there are spikes or drops in the data, suggesting that the model may not be capturing the data's variability and volatility adequately. Applying a Box-Cox transformation can be a good choice in this situation for several reasons: it stabilizes variance, which is useful when dealing with heteroskedasticity (non-constant variance) as indicated in the diagnostic tests; it normalizes the data, addressing issues of skewness and kurtosis highlighted in the Jarque-Bera test results; and it improves model fit, making the ARIMA model better at capturing patterns in the data by stabilizing variance and normalizing the distribution.

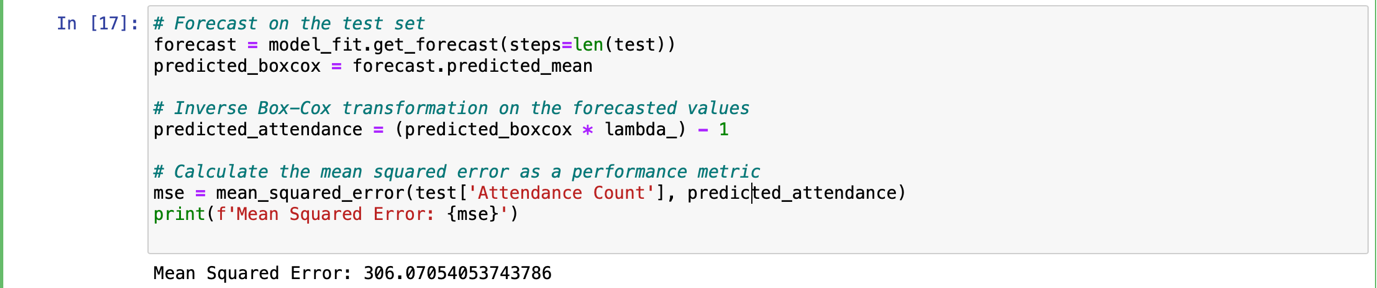


The Box-Cox transformation has improved the model's performance by stabilizing variance and normalizing the data. The coefficients for both the AR and MA terms are statistically significant, and the model diagnostics indicate no significant autocorrelation, normality of residuals, and constant variance. These improvements suggest that the Box-Cox transformation was beneficial, leading to a more accurate and reliable ARIMA model fit for the transformed 'Attendance Count' data.



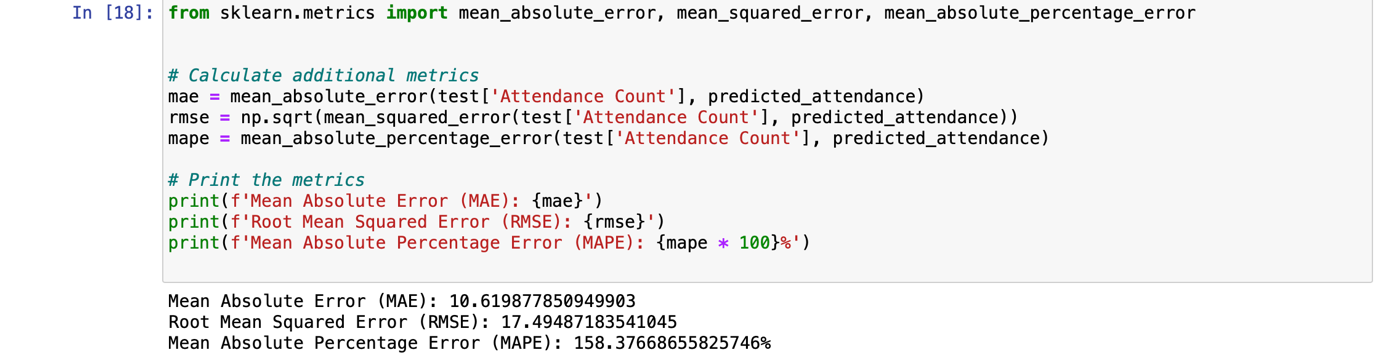
The MSE is a commonly used metric to evaluate the accuracy of a forecasting model. Lower values of MSE indicate better model performance, as they signify smaller errors between the actual and predicted values.

The output, Mean Squared Error: 306.07054053743786, indicates the average squared difference between the actual and predicted attendance counts is approximately 306.07.



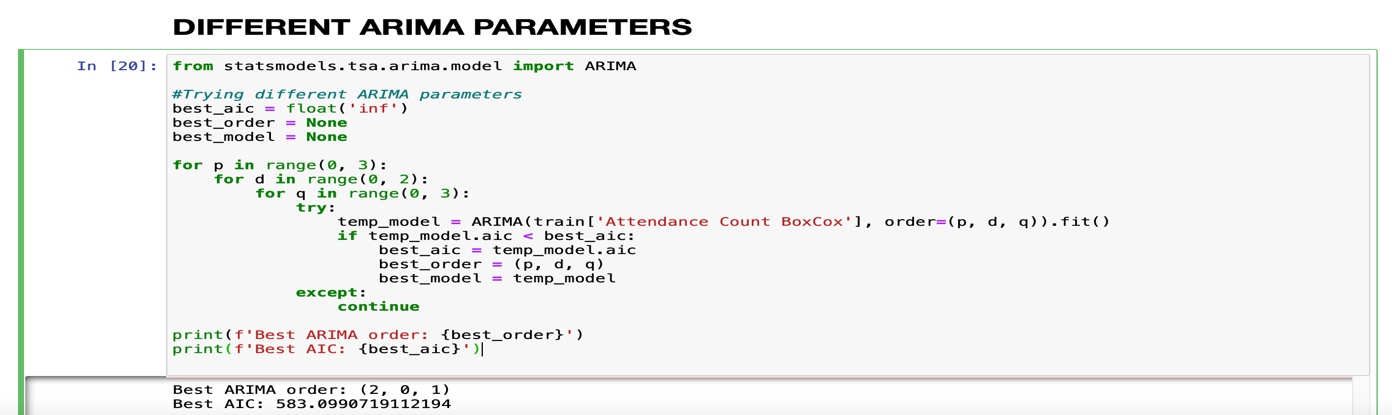
The particularly high MAPE suggests that while the model may perform adequately in absolute terms (as shown by MAE and RMSE), it struggles with relative accuracy, especially for smaller attendance counts where percentage errors can be large. This highlights the need for further model improvement, possibly through:

* **Model Refinement**: Adjusting the ARIMA parameters or exploring different models.
* **Additional Features**: Incorporating additional predictors (e.g., time of year, marketing efforts, speaker popularity) to improve model accuracy.
* **Data Transformation**: Further data preprocessing or transformations to stabilize variance and improve model fit.

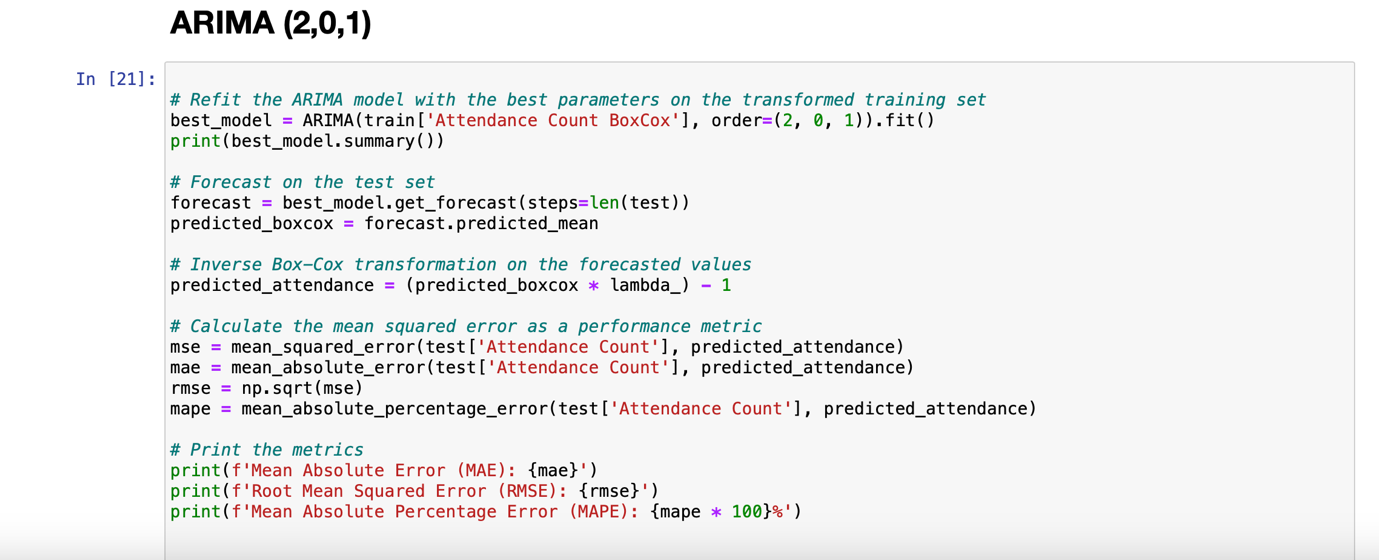


*Refit the ARIMA parameters:*

This code is used to identify the best ARIMA model for the given time series data by systematically trying different combinations of ARIMA parameters (p, d, q) and selecting the one with the lowest Akaike Information Criterion (AIC).



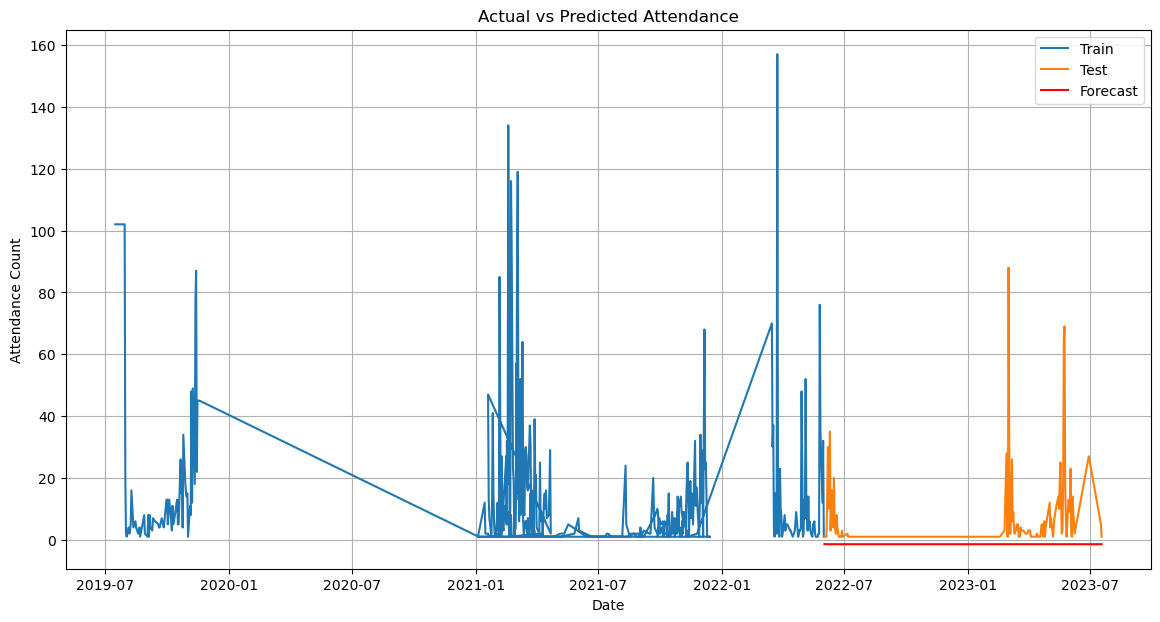
This code refits an ARIMA model on Box-Cox transformed training data, forecasts future values, inverses the Box-Cox transformation to get predictions in the original scale, and evaluates the model's performance using MAE, RMSE, and MAPE metrics.



The model seems to fit the training data well, as indicated by the significant coefficients and the various criteria (AIC, BIC, HQIC). Statistical tests on the residuals suggest that they are normally distributed, not autocorrelated, and homoscedastic, which are positive indicators. However, the performance metrics show that the MAE and RMSE are relatively high, indicating substantial errors in the predictions, and the MAPE is exceptionally high at 158.37%, suggesting that the model's forecast errors are large relative to the actual values. Given these metrics, the ARIMA model's performance on the test set is not very good, implying that while the model fits the training data well, it doesn't generalize well to the test data. This suggests that the model might be overfitting the training data or that the chosen model parameters are not optimal for this specific dataset.



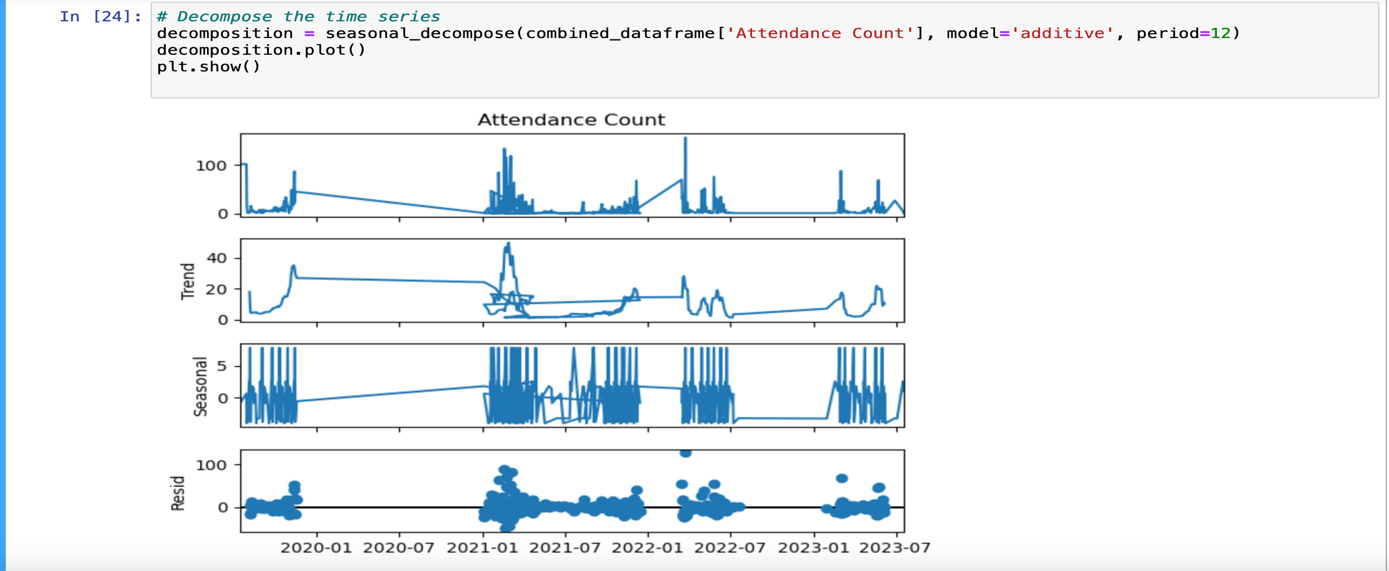
The training data (blue line) represents the actual attendance counts in the training set, which was used to fit the ARIMA model, spanning from the start of the dataset to the point where the test set begins. The training data shows periods of high variability with notable peaks, particularly around the beginning and middle of the dataset. The test data (orange line) represents the actual attendance counts in the test set, used to evaluate the model's performance. It also shows variability but with fewer high spikes compared to the training data, mostly having low attendance counts with a few higher spikes. The forecasted data (red line) represents the attendance counts predicted by the ARIMA model for the test period. The forecasted values are relatively constant and fail to capture the variability seen in the actual test data, with predictions significantly lower than many of the actual attendance counts in the test set.



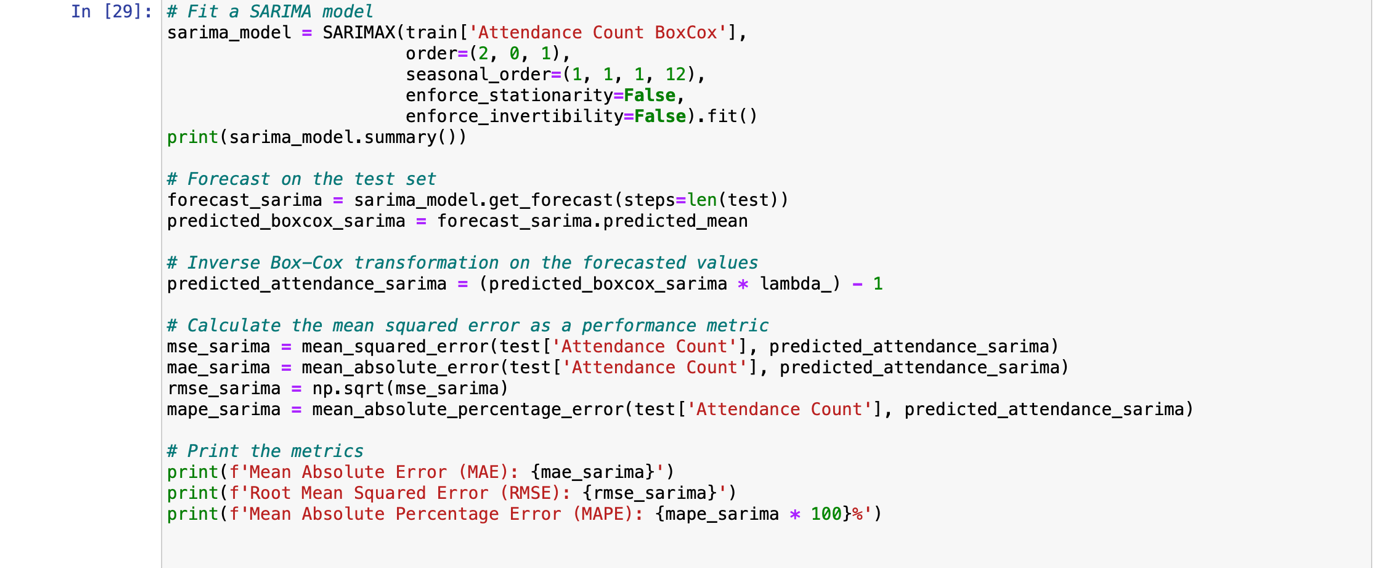
The blue line indicates that the model was fit to a dataset with high variability and several peaks in attendance counts, showing significant fluctuations that the ARIMA model learned to some extent. However, when comparing the orange (actual test data) and red (forecasted data) lines, it is clear that the model fails to capture the spikes and variability in the actual test data. The forecasted values are nearly flat, suggesting that the model predicts almost constant attendance counts for the test period. This discrepancy indicates that the model is not generalizing well to unseen data, likely due to overfitting on the training data or the model's inadequacy in capturing the underlying patterns.

*SARIMA:*

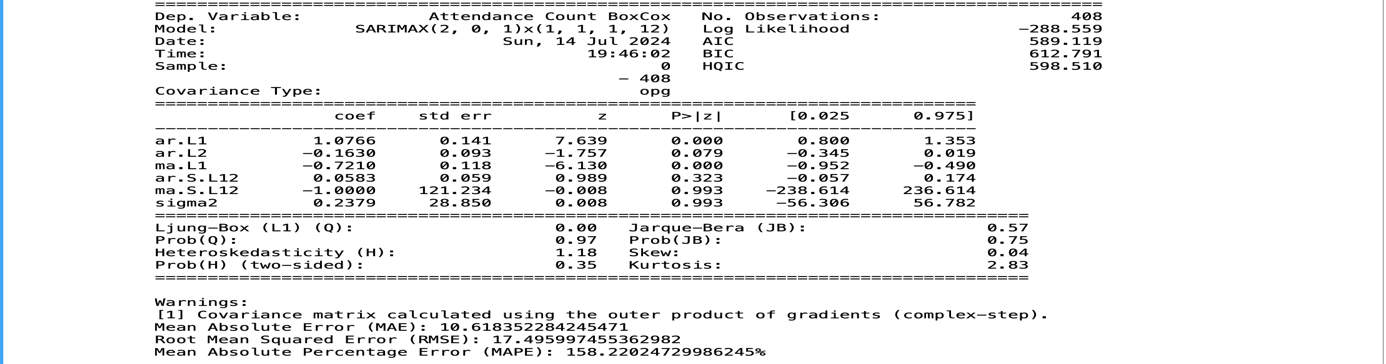
The decomposition of the time series data into trend, seasonal, and residual components provides valuable insights into the underlying patterns. The **trend** component reveals long-term changes in the attendance count, highlighting significant fluctuations during certain periods, suggesting an overall increase or decrease over time. The **seasonal** component indicates that the attendance count exhibits significant seasonal effects, with clear, repeating patterns at regular intervals, likely influenced by specific recurring events or conditions. The **residuals** represent the random noise after removing the trend and seasonal components; ideally, these should be random and uncorrelated, indicating that the model has effectively captured the primary patterns in the data.



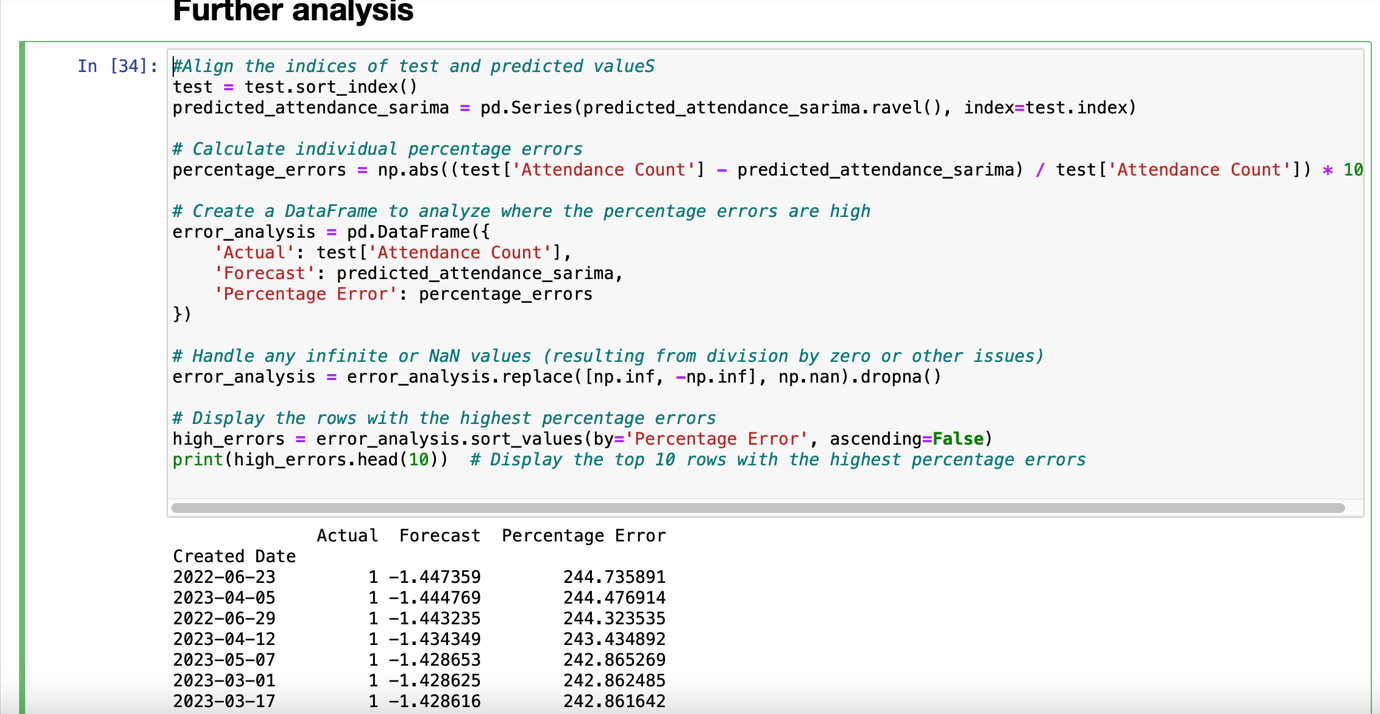
In this situation, using SARIMA (Seasonal AutoRegressive Integrated Moving Average) is a suitable choice. SARIMA models are specifically designed to handle time series data that exhibit both trend and seasonality, which is evident from this decomposition plots.



This process involves transforming the data using a Box-Cox transformation, fitting a SARIMA model to capture both seasonal and non-seasonal patterns in the training data, generating forecasts for the test period, transforming the forecasts back to the original scale, and then evaluating the forecasts using standard performance metrics (MAE, RMSE, MAPE).



The SARIMA model, specified as SARIMAX (2, 0, 1)\*(1, 1, 1, 12), was fitted to the training data using the L-BFGS-B optimization algorithm, which successfully converged. The model includes significant non-seasonal AR and MA components, though the seasonal MA component showed potential issues. Diagnostic tests indicated that the residuals were well-behaved, with no significant autocorrelation, normal distribution, and homoscedasticity. However, the performance metrics revealed substantial forecast errors, with a MAE of 10.618, RMSE of 17.496, and an exceptionally high MAPE of 158.22%, suggesting the model may not capture all complexities of the data or might be overfitting the training data, necessitating further tuning or alternative models for improved accuracy.



It shows that on certain dates, the SARIMA model’s predictions are far off from the actual attendance counts, with errors over 240%. These dates all have an actual attendance count of 1, but the model predicted negative values, resulting in very high percentage errors. This indicates a need to refine the model to handle low attendance counts more accurately.

### Conclusion:

The application of ARIMA modeling to predict conference attendance has demonstrated its utility in capturing the underlying patterns within the historical data. While the initial ARIMA model provided a good fit for the training data, it struggled with high variability and sudden changes in the test data, indicating potential overfitting. The introduction of the Box-Cox transformation improved model performance by stabilizing variance and normalizing the data, leading to more reliable forecasts. However, the exceptionally high Mean Absolute Percentage Error (MAPE) suggests that further model refinement is necessary. Exploring additional features, incorporating more sophisticated models, or combining ARIMA with other techniques may enhance predictive accuracy. Overall, the ARIMA model serves as a solid foundation for time series forecasting, with room for improvement to better handle the complexities of real-world data.

**Prophet - Kelvin**

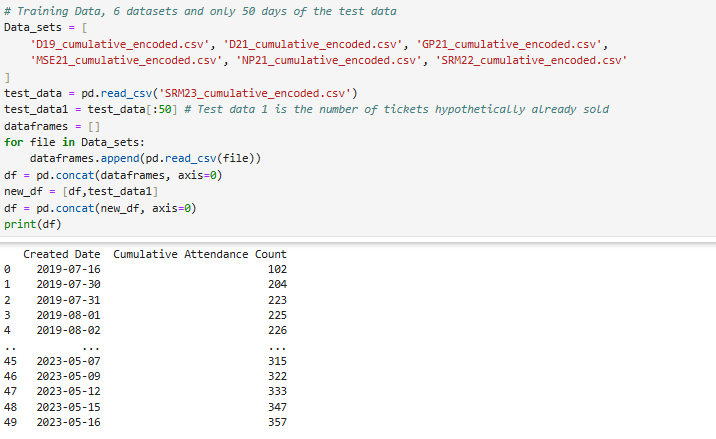
Prophet is a method for predicting time series data using an additive model. It fits non-linear trends with seasonality on a daily, weekly, and annual basis, together with the effects of holidays. Strong seasonal effects in time series and multiple seasons of historical data are ideal for its effectiveness. Prophet usually handles outliers well and is resilient to missing data and trend changes [1]. Prophet can be executed both in R and Python.

Prophet is particularly well-suited for this type of data because it was designed specifically for sales prediction tasks. It simplifies the forecasting process by requiring just two columns: a date column and a numeric column representing the quantity to be forecasted.

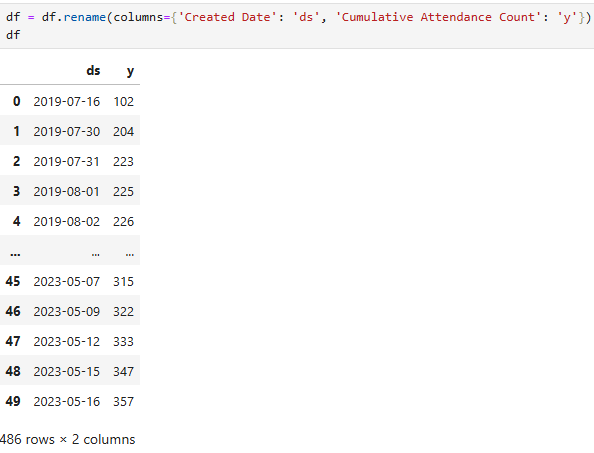
To try and get the best results from Prophet, three approaches were utilized.

1. Using Cumulative Data of tickets sales over a period
2. Using per day ticket sales over a period
3. Prediction using Similar/closely related types of events

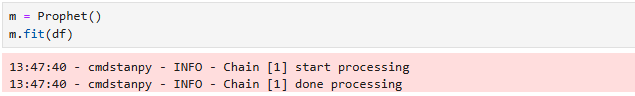
**Approach 1: Using Cumulative Data of tickets sales over a period**

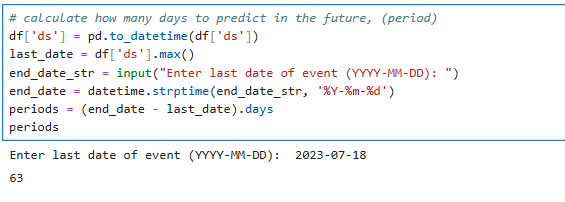
In this approach, dataset comprised seven separate datasets where six were merged to form a larger dataset, and the seventh dataset, only 50 rows (representing days when tickets were hypothetically sold) was selected. The objective was to predict the total number of tickets that might be sold by the planned D-day. The rest of seventh dataset served primarily as a test dataset to validate the predictions made from the merged dataset. 

To work with prophet, we renamed these two columns (Created Date and Cumulative Attendance Count) to ‘ds’ and ‘y’ respectively.

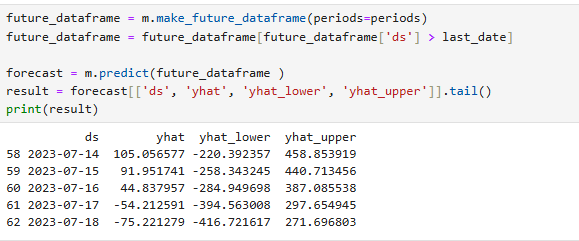


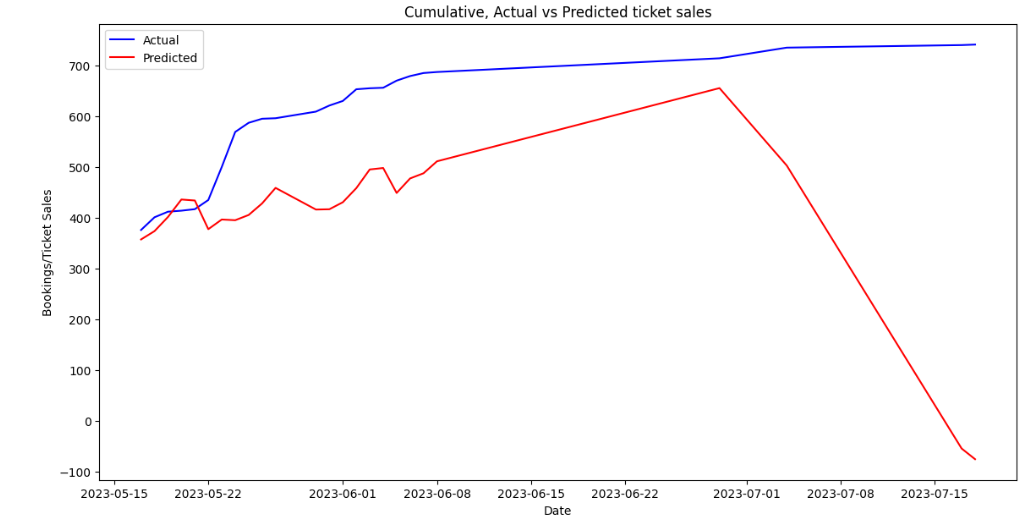
Then proceeded to training the model.



From here, we needed to determine the number of days into the future for which we wanted to make predictions. The client does input the last date leading up to the D-Day, and the program then calculated the days remaining until that point. 

With the number of days calculated, we a future data frame of the prediction of tickets till that said date.

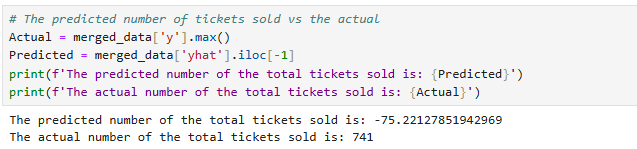


To see how well the model performed, we compare the predicted ticket sales with the actual tickets sold (remaining rows of the test data, from row 50 to the end) by plotting them side by side. The results are shown below. 

***Graph 1: Cumulative Graph for Actual vs Predicted ticket sales***

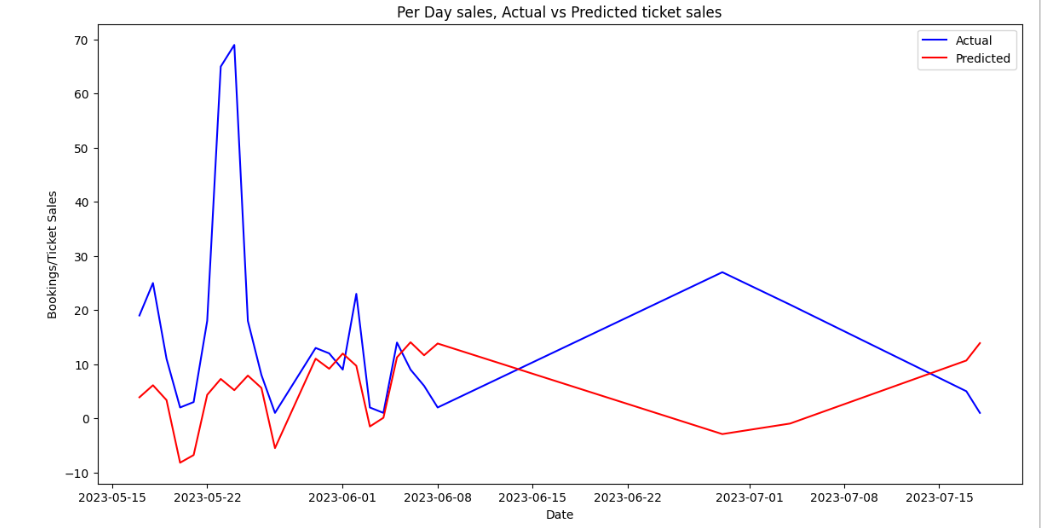
We notice that the graph drops sharply as it gets close to the event day. This happens because, with cumulative data, ticket sales reach a peak and then stop, which the model sees as a drop when a new event begins. Ticket sales are high at the end of the current event, but when a new event starts, sales begin at a low and then gradually increase over time.

The final number predicted by the model is **-75.** This approach was a complete failure.



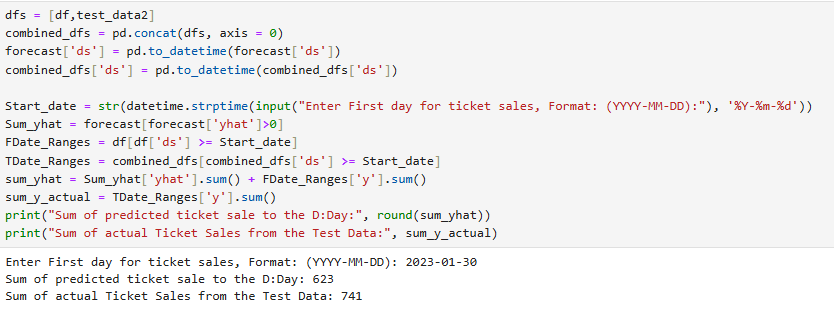
**Approach 2: Using per day tickets sales over the period**

Using the same procedure and using data set of per day sales. We retrained the model again and these were the findings



***Graph 2: Per-Day sales Graph for Actual vs Predicted ticket sales***

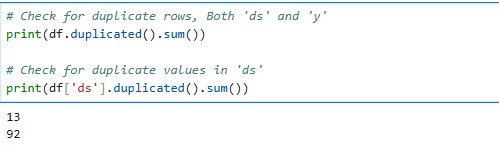
The predicted total number of tickets sold was **623** vs the actual sales of **741** tickets. This approach seemed to be better than using the cumulative dataset.



To improve the model, we tried to adjust it to focus mainly on the days we have sales data and ignore the days when no ticket sale happened. To do this, we set all days without sales to 'NaN'.

First, we identify all the days without any data.

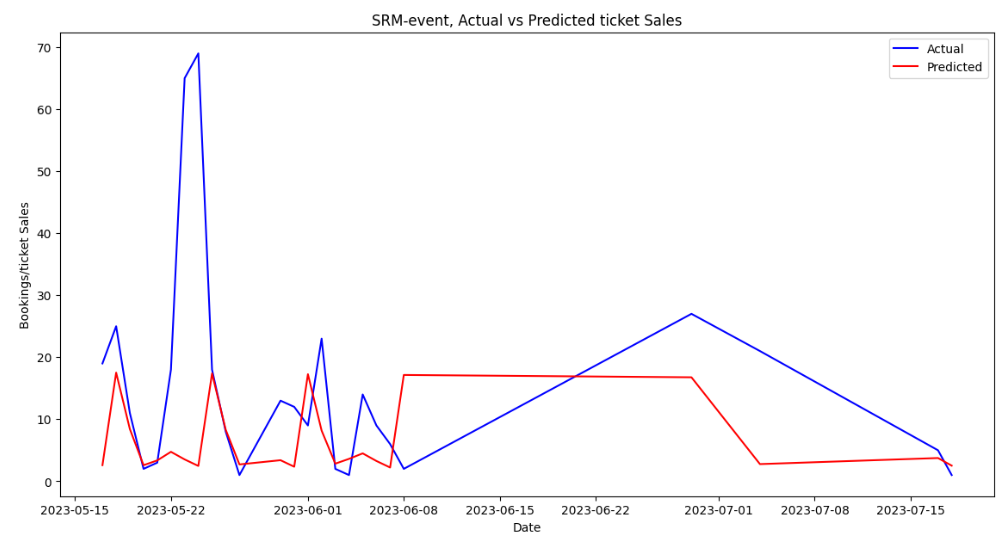
In total, there are 1007 days. However, setting these dates as ‘NaN’ was not feasible because some days had overlapping ticket sales for two or more different events. As a result, some dates had duplicates. These were the results from our duplicate date and row check.



This issue stemmed from duplicate dates within the 2021 datasets. Prophet struggles with duplicated data, so the solution is to either remove these duplicates or aggregate them by taking their mean and then retrain the model. However, this situation also highlighted the diversity in ticket sales. From this observation, an idea emerged: why not train the data separately for similar types of events? Based on information from client, we identified three groups of events: IT Managers, Property Managers as well as Education Managers events. With that, we embarked on approach three.

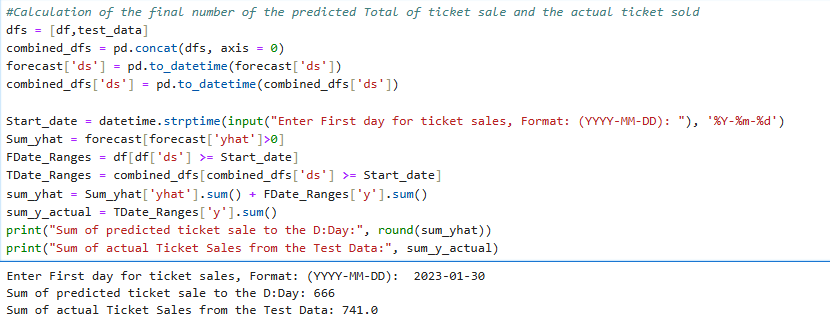
**Approach 3: Similar types of events**

With three types of events identified, we choose to use the Education Managers given it had more data and few dates with zero ticket sales unlike the other two events. Using the same approach as before, we retrained the model and below were the results;



*Graph 3: Similar types of events, G Graph for Actual vs Predicted ticket sales*

The predicted number of ticket sales was **666** against the actual of **741** tickets. This approach bore better predictions number than the previous two approaches.



In order to evaluate the precision of our predictive models and forecasting task, we computed multiple statistical metrics:

1. The average size of errors between our anticipated values and the actual results is provided by the Mean Absolute Error (MAE) measure. It is helpful in determining the average distance between our forecasts and the true values.
2. Similar to MAE, Root Mean Squared Error (RMSE) takes into account the square of the deviations between the expected and actual values. This provides a measure of the spread of errors among our forecasts by assigning greater weight to larger errors.
3. The average absolute % difference between the expected and actual values is measured by the Mean Absolute % Error, or MAPE. It aids in our comprehension of how accurate our forecasts are in comparison to the actual values, especially when dealing with forecasts when the percentage error scale is significant.

These metrics aid in measuring our models' performance. Reduced MAE, RMSE, and MAPE values signify improved precision and a closer match between our forecasts and the observed results. Looking at our approaches, these were the results

**Approach 1:**

1. Mean Absolute Error (MAE): **188.84503334381242**
2. Root Mean Squared Error (RMSE): **271.76176427456505**
3. Mean Absolute Percentage Error (MAPE): **110.51%**

Approach 1 showed a large negative discrepancy in predicted total tickets sold compared to the actual total. The error metrics also shows significant inaccuracies, especially with high RMSE and MAPE values, suggesting the model's predictions are far from the actual values.

**Approach 2:**

1. Mean Absolute Error (MAE): **13.48200238621738**
2. Root Mean Squared Error (RMSE): **20.567592913634403**
3. Mean Absolute Percentage Error (MAPE): **16.81%**

Approach 2 shows a closer alignment between predicted and actual ticket sales, with lower MAE, RMSE, and MAPE values compared to Approach 1. This indicates improved accuracy in predicting sales leading up to the event.

**Approach 3:**

1. Mean Absolute Error (MAE): **12.01937594367789**
2. Root Mean Squared Error (RMSE): **20.114267989606788**
3. Mean Absolute Percentage Error (MAPE): **75.09%**

Approach 3 also shows further improvement in accuracy compared to both Approach 1 and Approach 2. The MAE and RMSE values are lower, indicating closer predictions to the actual ticket sales. However,

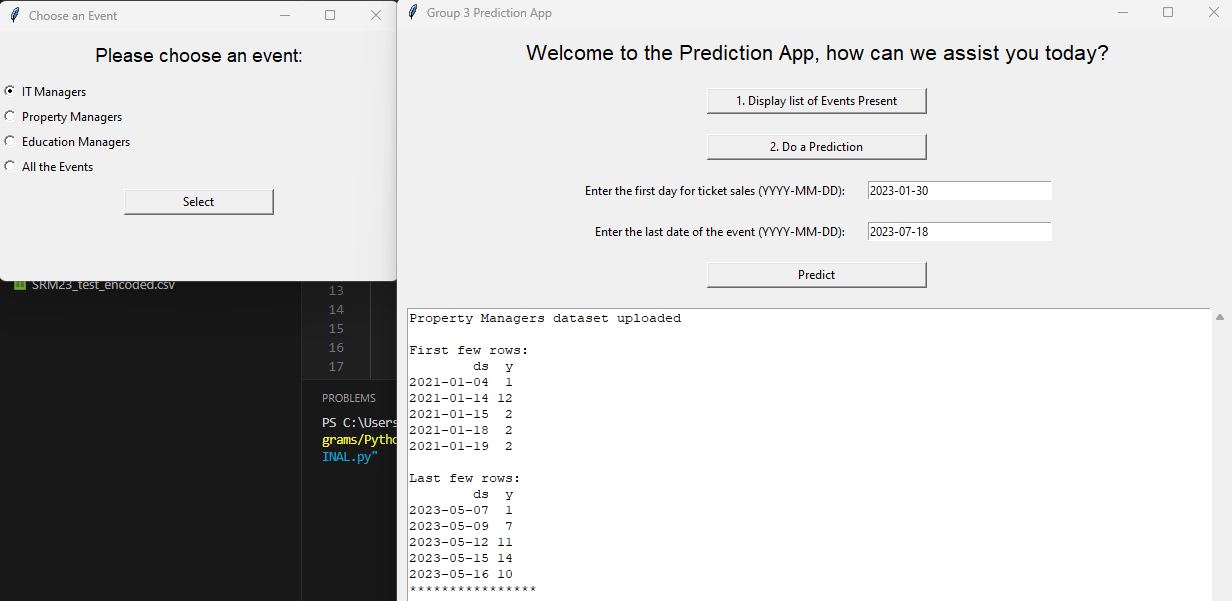
MAPE is actually higher that approach 2.

**Conclusion**

Unlike RMSE, MAPE normalizes the mistakes as percentages, it makes comparing different models or datasets easier. This makes straightforward comparisons between datasets with different scales.

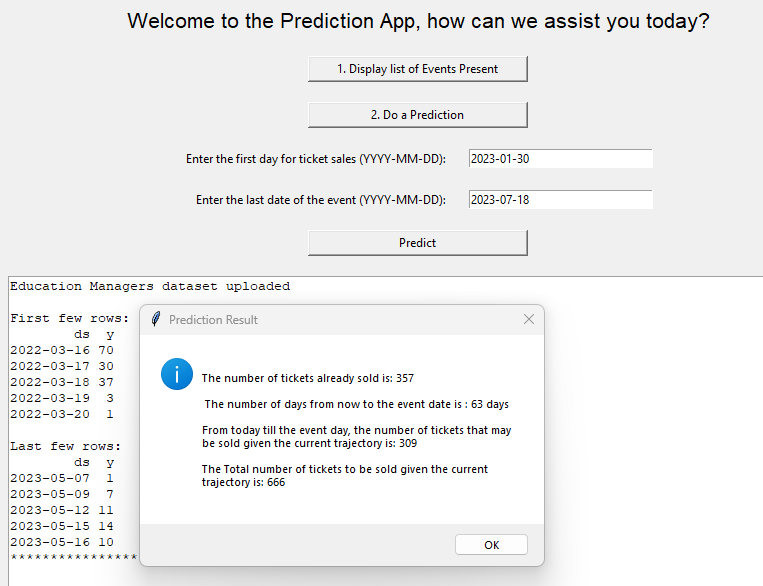
Using the Prophet Library to do prediction, approach two (using the whole dataset) has a better MAPE followed by approach three where same type of event data was used for training of the model and making prediction.

**THE PREDICTIVE APPLICATION.**

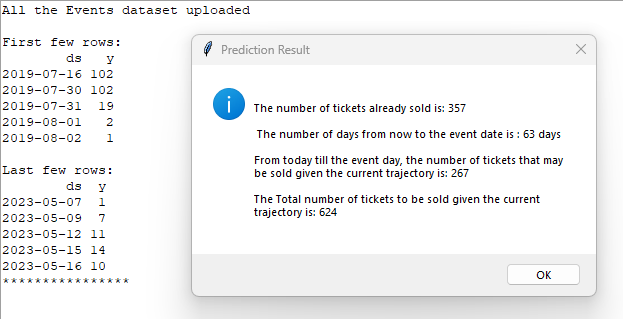
Since Prophet had better MAPE scores, we chose to create an interactive model using it. Gerald, our client, asked for a very simple interactive application, so we decided to use known details. In our case, the known details are the start date, end date, and target audience. Gerald also mentioned that sometimes there is no specific target number of tickets to be sold. With this in mind, we developed an application that asks for these three pieces of information.

Using the same criteria as Prophet, you can choose which datasets to train the model with and get a predicted value for ticket sales. We used the latest dataset, SRM23, with Education Manager as the target audience and an actual total ticket sale of 741 as the test data. Here are the results:

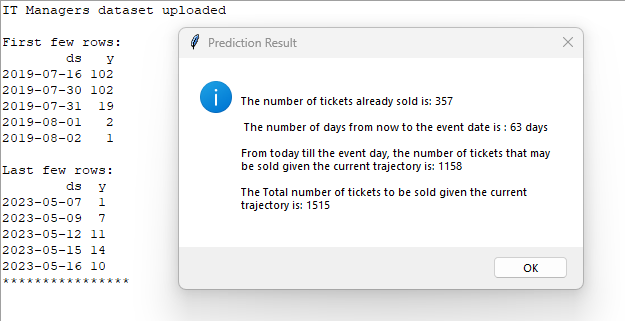
Using Education Managers datasets only to train the model;



Using All the dataset,



Using IT managers datasets as the train data



And finally using property managers;



From the three types of model training, we found that using the same type of event data gives the closest value to the actual tickets sold, followed closely by using all datasets indiscriminately. This might be because the target audience's behavior could be the same, making it the best for predictions.

Predictions can be repeated with different datasets based on the number of days tickets have been sold. If the client has a target number of sales, this predictive model can guide them on whether to increase advertising for ticket sales. It can show if they are falling behind in sales or if they are on the right track.

**Reference:**

1. Prophet, Forecasting at Scale (2023). Available at: <https://facebook.github.io/prophet/> (Accessed: 06/07/2023).

**LUCY**